Implementation of Bayesian Updating and Markov Chains to Improve Simulation of Stochastic Processes in Construction

by

Michael Steven Samuel Werner

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science
in

Construction Engineering and Management

Department of Civil and Environmental Engineering
University of Alberta

Abstract

This thesis was mainly concerned with improving model-based prediction and forecasting of state variables of processes or systems using new, periodically-generated data within the construction domain. Some of these data are generated from field tests performed during feasibility studies, while other data are acquired during the execution phase of the project. The primary matters that are examined in this research relate to prediction of a physical attribute of the environment that is interacting with the work face; specifically, ground conditions in tunneling projects. The approaches explored include the use of Markov chains and statistical distributions for representing uncertainty and stochasticity within simulation models in a dynamic fashion. Parameters for these are changed frequently, using Bayesian updating algorithms, as new data are received. The simulations and updating were implemented in an automated fashion. The merits of these techniques were demonstrated in case studies implemented within the tunnel construction domain. Results demonstrate that the proposed method was able to provide more reliable predictions with relatively little effort.

Preface

This thesis is an original work by Michael Werner.

Chapter 3 is adapted from work published as L. Zhang, R. Ekyalimpa, S. Hague, M. Werner, and S. AbouRizk, “Updating geological conditions using Bayes’ Theorem and Markov chain,” in *Proceedings of the 2015 Winter Simulation Conference*, pages 3367 – 3378. It is reprinted with permission of the Institute of Electrical and Electronics Engineers. Figures 3.2, 3.3, and 3.4 are reprinted with permission of SMA Consulting Ltd.

Acknowledgments

I would like to thank and acknowledge the guidance and contributions of my supervisor Dr. Simaan AbouRizk. Thank you for all of the support and insight allowing me to reach this milestone in my academic career. Your boundless knowledge in construction engineering and simulation is remarkable and I have been so lucky to be able to learn from you in this domain.

Special thanks to Post-Doctoral advisors Dr. Ronald Ekyalimpa and Dr. Limao Zhang for their helpful collaboration and support in gaining a better understanding of simulation modelling and analytical techniques.

I would like to express thanks to Stephen Hague, Programmer Analyst in the Construction Engineering Department for his guidance and assistance in the implementation and use of the Simphony modeling environment. Your willingness to guide me in the software and computing aspects of this work was much appreciated.

Particular gratitude to the City of Edmonton Drainage Department and SMA Consulting Ltd. for providing project information and data and allowing on-site research to take place.

I would like to thank all of my family and friends for their continuous support throughout my graduate studies journey. Lastly this work would not have been possible without my wife Jenna, always by my side and having endless patience and understanding.

Table of Contents

List of Tables

List of Figures

List of Appendices

CHAPTER 1

INTRODUCTION

1.1 Background

Many real-world systems, such as weather, biological processes, and economics, are stochastic and dynamic in nature. A stochastic, dynamic process is one that changes randomly or unpredictably and in a manner, that is not trivial. This contrasts a deterministic process, which displays no randomness and will produce the same result from a given starting point. Due to the unpredictability of a stochastic process, such systems are appropriate for analysis using simulation-based methods. In doing so, the process can be modelled to reproduce the randomness using probabilities and distributions.

There is a subset of these systems that are characterized by sequential features. This is a system that proceeds sequentially but remains stochastic and dynamic in nature. Examples of this type are many construction processes, where a process is a sequence of interrelated events that may repeat. Envision the construction of a road, which is comprised of many steps and events that each takes a certain amount of time. Each event, or step, in the process is completed in sequence, and each step may take a variable amount of time to complete thus presenting some randomness. Overall, the process is sequential and repetitive in nature, and the outcome of the process, in terms of road construction duration, depends on many factors.

To analyze these complex systems, some of which are existing, others are developed to meet special requirements for the system being analyzed. An efficient way of making use of these algorithms has involved embedding them within a simulation environment(s).

The following subsection will introduce the various types of algorithms and simulation component techniques that were used to achieve the objectives of this thesis. Simulation models coupled with these algorithms were implemented on selected case studies to demonstrate the practical and theoretical contributions of these specialized models within the construction domain.

1.1.1 Markov Chains and Hidden Markov Models (HMM)

Hidden Markov models (HMM) can be used to represent a time-based sequence of events, where a parameter or characteristic passes through states or nodes (Rabiner, 1989). At each state or node, certain observations or signals are experienced. Hidden Markov models attempt to model the intricacies of such a system to predict how state transitions and observations will occur as well as the probabilities with which these arise based on the current state and information. These techniques have been found to be useful when applied to modelling of tunneling ground conditions (or geological conditions, which are used interchangeably in this thesis) and were, therefore, used in this thesis.

1.1.2 Statistical Distributions

A statistical distribution is a way of representing information with the uncertainty that surrounds this information. It can be thought of as a set of data, either observed or theoretical, that is organized according to frequency of occurrence. Distributions can follow many forms and types, and distribution fitting, itself, is a notable research area in the field of statistics. Simulation modelling relies heavily on statistical distributions to sample data for the purposes of predicting the intricacies of processes and systems. Statistical distributions can be either discrete (integer) or continuous (non-integer) in terms of their data. Typically, in the domain of construction, continuous distributions are better suited for modelling durations due to their flexibility.

Statistical distributions were fit from data of past tunnel projects and were used to sample the time required to complete various activities in the tunneling process, such as train travel, liner loading/unloading, and, most importantly, excavation. When data were absent, we subjectively fit distributions using input qualitative information. These distributions were then used as inputs for simulation modelling of the tunnel and were used as the prior distributions for Bayesian updating techniques.

1.1.3 Simulation Modelling

Simulation modelling can be thought of as the development, experimentation, and use of a scientific abstraction of a physical system or process (Banks, 2001). A simulation model acts a prototype to predict, analyze, and assess an actual physical system in the form of a computer based representation. While the origins of computer simulation modelling date back to the 1940s and 50s, it was not until the 1980s and 90s that simulation modelling became widely-accepted and used. Challenges associated with inadequate computing power, personnel expertise, and cost greatly limited the early use of this technique. However, as the commercial availability of powerful software, such as GPSS and SLAM developed by Alan Pritsker and associates, arose, simulation modeling began to grow in popularity, now becoming widely-utilized. The advantages of a well-built simulation model are that it can assist in reliably predicting the future performance of real-world systems, allowing engineers and scientists to make decisions and plan around the results. Simulation modelling can be applied to many facets of science, environments, and real-life systems.

1.1.3 Bayesian Updating

The primary function of Bayesian Updating is to improve the legitimacy and accuracy of predictions through the incorporation of newly acquired information. For example, an initial prediction or a probability of certain outcomes is known. As additional information is obtained about these outcomes, the strength of the predictions and probabilities of such outcomes can be improved. Bayesian updating is reliant on collected data of a real system being studied, and the data are input into algorithms that improve prediction models being used. As there is sufficient time for corrective action, long-term projects lend themselves to this type of updating and will benefit the most from this technique. While there are many algorithms and techniques that can be used in Bayesian updating, the overarching themes of these is the adjustment of the posterior or future probability based on prior probabilities and actual observations. In this thesis, Bayesian updating was used specifically to update the statistical distributions and the parameters of a Hidden Markov model (HMM). Bayesian updating of normal distributions has been applied, previously, in tunneling construction, where it was shown to be effective at improving simulations using actual construction data (Chung, 2006). A specific algorithm, named the Baum Welch algorithm (Baum and Petrie, 1966), was used to update the transition and observation probabilities of an HMM model. This algorithm is a specific type of Bayesian updating process known as an Expectation Method or EM method.

1.1.4 Case Study

Case studies were explored in this thesis to demonstrate the merits of the above techniques and their use in construction management and simulation. The specific domain of tunneling was selected due to the sequential nature of the process and the level of uncertainty surrounding ground conditions that is often encountered. Past personal experience in tunneling construction, as well as previous papers on the topic, assisted in better understanding the value of improvements in predictions and the level of uncertainty around such predictions traditionally. The case studies focused on demonstrating how the statistical distributions used for modelling process variables were updating using the proposed method as well as on demonstrating the ability of the model to improve ground condition prediction in tunneling. For clarity, the term ground condition or geological condition is used, interchangeably, in this thesis to refer to the material that is being excavated in a tunnel project.

1.2 Problem Statement

1.2.1 State of Practice (Industry Practice)

An overarching problem that is predominant in both academia and practice is the failure to make use of simulation models, new data, and updating algorithms in a holistic and harmonious fashion to generate more accurate predictions and forecasts.

Information and data that are available during the planning and early design phase of construction are often not representative of the real system. Commonly, data sets are made of data that are outdated, inconsistent, or incomplete. Many times, data are assumed from past projects or are collected from domain experts, which can lead to issues regarding data accuracy. This is especially prevalent in geotechnical investigation. Typically, geotechnical investigations consist of intermittent borehole samples taken across the alignment (in the case of tunnel construction). While the data obtained directly at the boreholes is relevant, there is no real data for areas between boreholes. While increasing borehole sampling frequency may increase the comprehensiveness of the data, taking boreholes at distances that would create a complete data set is extremely costly and impractical. Rather, the current practice for borehole sampling focuses on providing a general overview of the ground conditions that are to be expected and to identify any major geotechnical risks, such as voids, water pockets and unexpected ground conditions. Especially in underground construction, the reality of the geotechnical profile is highly uncertain due to the inability to observe the entirety of conditions underground on many projects, where only certain portions of the project may be investigated.

Overall, the state of practice in Construction Engineering for developing project plans and designs for tunneling projects is often based on assumptions, intuition, personal experience, and incomplete data. However, as a project progresses, new information, in particular regarding conditions underground, becomes available: if a new geotechnical condition is found, or a better understanding of the geotechnical profile is realized, construction operations and plans should reflect the new information. A method capable of enhancing the reliability of original predictions through the incorporation of newly acquired data from the actual system is necessary.

1.2.2 State of Art (Academic Practice)

Current use of Bayesian updating is limited to (1) the updating of Markov chains and (2) the updating of statistical distributions. There has been some work using Bayesian updating of Markov chains for updating geological conditions for tunneling at MIT (Ioannou, 1989). Other work involved similar methods applied to mountain tunneling and excavation work (Guan, 2012). The updating of statistical distributions has been done for Normal distributions (2 parameter distributions) (Chung, 2006).

Two gaps are evident: first is the lack of integration of Bayesian updating with an isolated simulation model and second is the absence of the application of techniques, such as HMM, in construction.

1.3 Study Objectives

1.3.1 Main Objective

Overall/Main Goal and Objective: The overall objective of this work is to use analytical and numerical methods, such as Bayesian updating, Markov chains, and discrete event simulation, to improve the way tunnel construction processes are modelled.

1.3.2 Specific Goals and Objectives

1) Identify, from past studies, the most efficient algorithm for predicting the state of systems/processes that vary sequentially
2) Develop prototypes for updating the techniques used for predicting the sequential state of ground conditions in tunneling
3) Develop prototypes for updating the techniques used for predicting statistical distributions representing specific variables in tunneling construction
4) Implement case studies to test and demonstrate concepts for the above objectives
5) Develop more reliable predictions of project durations using the proposed methodology

1.3.3 Expected Research Contributions

- Industrial Contributions
  - Effective and proactive use of data collected from site to enhance project delivery
  - Demonstration of how statistical distributions that represent the rate at which different TBM machines advance in different ground conditions can be generated and reported
- Academic Contributions
  - Exploration of various algorithms, from the computing science domain, which are capable of solving updating problems that exist in tunnel construction.
  - Integration of Bayesian updating and HMM techniques with simulation

1.4 Organization of Thesis

The following chapters detail the achievement of the previously stated research objectives and contributions.

Chapter 2 of this thesis presents a literature review into the work previously done in the fields of construction simulation, Bayesian updating and applied algorithms, Markov and hidden Markov modelling, and the use of statistical distributions in construction.

Chapter 3 introduces the use of Bayes techniques applied to update parameters of a Markov chain for which the geological conditions of a one-meter section of tunnel construction can be updated. Graphical representations are provided to display how predictions of the Markov chain change based on new information.

Chapter 4 details the Baum Welch algorithm, an applied Bayesian updating technique, and tests its implementation in the Simphony modelling environment.

Chapter 5 of this thesis details the implementation of hidden Markov models (HMM) in the Simphony modelling environment with dedicated modelling elements allowing training, updating of model parameters, and integration of HMM models into a simulation model.

Chapter 6 presents a tunneling case study demonstrating the effectiveness of the integration of Bayesian updating techniques and hidden Markov models (HMMs), their implementation in the Simphony simulation modelling environment, and their use for predicting project durations from ground condition data.

Chapter 7 presents the conclusion of this research, listing its contributions as well as its limitations. This chapter also offers some suggestions for future work.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

This section serves to summarize the background and overview of the techniques used in the research of this thesis. A brief history of the techniques is provided, which is followed by high-level discussion of the theories.

2.1.1 Statistical Distributions

In the world of modelling, there are various ways that uncertainty can be represented, including probability and statistics or fuzzy logic. The focus of this research uses statistical distributions, since most of the simulation applications we employ in the thesis are Monte Carlo-based techniques. Statistical distributions can be in the form of discrete distributions, where only integer values are represented, or a continuous distribution, where any value can be observed. In modelling construction processes, continuous distributions are more commonly used, as the values being modelled typically represent continuous parameters such as durations, costs, and rates (e.g., production rates, penetration rates, travel rates). Certain distributions, such as the beta, normal, exponential, and Weibull distributions, are commonly used for modelling construction operations.

Between 1700 and 1900, there was significant growth in statistics as it spread from its roots in astronomy and geodesy to psychology, biology, and the social sciences. In 1733, De Moivre's work in approximating the term of a binomial expansion yielded the publication that detailed what is now known as "the binomial distribution." Certain approximations and work furthered by Karl Pearson in 1926 led to the first description of the normal distribution. (Grinstead, 1997)

According to Karl Bury, a distribution function can be defined as the uncertainty aspect of a random variable. What form the distribution takes is dependent on the parameters of that given distribution function. The sampling of distributions provides an assessment of the statistical uncertainty for this variable (Serfozo, 2009).

As mentioned above, statistical distributions come in two primary forms, namely discrete and continuous distributions. In a discrete distribution, only integer values are possible. In construction engineering, it is commonly used to describe variables such as quantities of equipment. The distribution is built upon the possible outcomes and the respective probabilities of those outcomes. Examples of commonly used discrete distributions are the Bernoulli, binomial, geometric, and Poisson distributions.

A continuous distribution cannot be defined based on specific integer outcomes. Rather, it is defined based on a range of values. Sampling the distribution can yield any value in this range to a certain level of significance. The shape, constraints, and density of the distribution depend, again, on the parameters of the distribution (Stamp, 2015). Some examples of commonly used continuous distributions in construction are the uniform, triangular, pareto, beta, and exponential distributions.

The definition of customized statistical distributions is sometimes required and provides notable advantages. If an adequate amount of information regarding the variable of choice is available, you may be able to define and fit a distribution around a data set, provided that the source of the data are legitimate and accurate (i.e., a sufficient volume of data, provided by experts). A value can be extracted from a distribution through a technique called sampling. This involves directly sampling a value from the given distribution. In the construction simulation application, sampling of a distribution provides model inputs for certain variables. For example, during the simulation of an earthmoving operation, the duration for excavating a certain amount of material is sampled from a distribution. For each iteration of the excavation process, a new duration is sampled.

A statistical distribution can be represented either mathematically or graphically. A mathematical representation defines a function that details the distribution, which contains the statistical parameters of that given distribution. The number and types of such parameters is highly dependent on the nature of the distribution. For example, a common and well-known parameter of many distributions is the mean, which represents the central tendency of the distribution. Other common parameters are shape factors (which define how the distribution is shaped), upper and/or lower bounds, standard deviation, or degrees of freedom. In general, the statistical parameters of the distribution define the characteristics of the distribution.

Alternatively, a distribution can be represented graphically. An advantage of this form of representation is the ability to visualize the shape and nature of a distribution. There are two common graphical representations of a statistical distribution, namely the visualizations of the

Probability Density Function (PDF) and the Cumulative Distribution Function (CDF). The PDF is the density of probabilities for the given values of the random variable, where probability density or likelihood is plotted against the set of values within the bounds of the distribution. The CDF is defined as the probability that a value, x, is less than a given percentile, P. The CDF is a plot of the cumulative probability versus the variable outcomes.

The fitting of distributions to a data set is an important technique for representing uncertainty. Four primary techniques are used to fit distributions: the regression method or parametric methods including the method of moments, method of L-moments, or the maximum likelihood method (Law, 2007).

The method of moments technique matches sample moments with theoretical moments as defined by equations for distribution parameters, such as the mean and standard deviation. To solve for the equation parameters, the first sample moment is equated to the first theoretical moment. The same process is then followed until n equations to solve for n parameters are obtained. These parameters then formulate the fitted distribution. The method of L-moments is analogous to the method of moments, but uses L-moments. These are defined as linear combinations of order statistics that can be used to estimate descriptive statistical parameters, such as standard deviation, mean, and skewness. The maximum likelihood method is a technique that attempts to maximize the likelihood function for certain parameter values to estimate the parameters of a statistical distribution for a data set. It attempts to maximize the probability of data to be under a given distribution defined by said parameters. Finally, the regression method estimates a distribution by finding a linear relation between the transformation of a cumulative distribution function and the data set.

The use of statistical distributions in construction and simulation modelling is extensive and these three topics often work together harmoniously. Many academic papers have demonstrated the use of statistical distribution in these field (Chung, 2006; Scheffer, 2014; Dang, 2013; Rahm, 2013; Werner, 2015), where system parameters are sampled from statistical distributions that capture the uncertainty in key construction variables, such as duration and production rate.

2.1.2 Markov Processes and Chains

A Markov process is a mathematical model developed by Russian probability theorist A. A.

Markov. Markov processes allow the analysis of certain complex systems that behave randomly over time. A Markov Process is a special type of discrete-time stochastic process, whereby the probability of going from one state to another at a future time depends only on the current state occupied at the current time. This is known as the Markov property and indicates that our ability to predict the state of a system at any time cannot be enhanced by the knowledge of values of earlier system states (Gillespie, 1992). If the number of states is finite, the process is called an absorbing Markov Chain. Moreover, if the transition can occur only at discrete points in time, the process is known as a Discrete Markov Chain.

A system can be classified as a Markovian process, so long as it has the following properties (Serfozo, 2009):

- A finite number of states
- An initial probability distribution
- The transition to a new state depends only on the current state (known as the Markov property)
- The transition probability between any two states is constant or does not change over time

The last property warrants some discussion as to its applicability to construction projects. Typically, the transition of a state does, in fact, change over time as a function of progress, conditions, materials, et cetera. Since this property would commonly be contravened within a construction context, a variation of a Markov process is required. This variation is known as a Dynamic Markov Chain, where the transition probabilities vary with time, thereby allowing the application of modelling performance forecasting.

Markov chains belong to a family of graphical models that are used in the analysis of dynamic, stochastic systems. They are comprised of states and their associated transition probabilities, with states representing possible values that a given state variable of the system can take on.

Markov chains are typically used for modelling sequential processes where a finite number of possible states are repeated. They have been applied in a variety of contexts, such as earth and atmospheric sciences (Stamp, 2015), speech recognition (Rabiner, 1989), computing processes (e.g., web searching, performance evaluation) and communication (Von Hingers, 2003), and chemical process engineering (Tamir, 1998).

In the domain of Markov processes and models, events related to a system can be mapped or related to the state of the system. The term “state” is preferred to “event” when describing Markov processes or models. The transition probability matrix of a process can be configured in such a way that the chain can be comprised of only one or a mix of state types. The first kind of Markov state is called an absorbing state but is also often referred to as a recurring state. An absorbing state is one that, if a Markov chain transitions into, it will never leave. Rather, the state continues to recur. States, in a Markov model, which are not absorbing states are referred to as transient states, since the chain can transition to all other states. Notably, this includes transitions to absorbing states. The state space for Markov processes can also be categorized as either discrete or continuous. A discrete state space for a Markov chain is comprised of a finite or countable set of possible states, while a continuous state space has infinite possibilities for state values. Markov chains are a special case of discrete-valued Markov processes. Consequently, for practical purposes, Markov chains will hereafter be regarded as discrete-valued.

2.1.2.1 Types of Markov Chains

Markov processes can be categorized based on the nature of the states they can take on, namely as a discrete-time Markov processes (DTMP) or continuous-time Markov processes (CTMP). Discrete-time Markov processes describe a process that can persist in a state for a predefined discrete measure and that has distinct boundaries between states (Norris, 1998). The discrete measure could be moments in time, physical distance, or any other appropriate discrete measure. In continuous-time Markov processes, changes to the system can occur at any time along a continuous interval (Norris, 1998). The persistence within a given state is typically modelled using continuous probability distributions. Although it is never regarded as an explicit parameter for a Markov process, the parameter used to track the persistence of a Markov process within each state is a crucial parameter, as it can be used to classify the process and facilitates Markov computations. Markov chain models are a special case of discrete-time Markov processes.

2.1.2.2 Traditional vs Hidden Markov Chains (HMMs)

At a very high-level, Markov chains can be placed in one of two categories: (1) hidden Markov chains and (2) non-hidden or traditional Markov chains. While hidden Markov chains are comprised of states (referred to as hidden states), transitions, observation symbols, and emissions, traditional Markov chains are only comprised of states and transitions. Both types have been extensively used for modelling different types of problems. In this thesis, a generic approach that involved presenting information and developing a simulation tool that lends itself to both hidden and traditional Markov chains was used.

Hidden and traditional Markov chain models can be represented in both a mathematical and a graphical format. The mathematical format makes use of matrices and vectors in the definition of the parameters of the Markov chain. The notation used for the Markov parameters, their corresponding matrices and vector, and their elements, based on Rabiner and Juang (1986) and Rabiner (1989), are summarized in Figure 2.1.

$S = \{s_1, s_2, ...s_T\}: A\ set\ of\ all\ possible\ hidden\ states$

$X = \{x_1, x_2, ...x_T\}: A\ set\ of\ all\ possible\ observable\ states$

$A = \{a_{ij}\}, a_{ij} = P(s_{t+1} = j \mid s_t = i): State\ transition\ probability\ matrix$

$B = \{b_{im}\}, b_{im} = P(x_t = m \mid s_t = i): Emission\ probabilities$

$\pi = \{\pi_i\}, \pi_i = P(s_1 = i): Initial\ state\ distribution\ or\ prior\ distribution$

$\theta = \{A, B, \pi\}: Markov\ chain\ \mathrm{mod}\,el\ parameters$

$U = \{u^{(A)}, u^{(B)}, u^{(\pi)}\}: Hyperparameters\ used\ to\ define\ the\ prior\ over\ \theta$

Figure 2.1: Markov Parameters and Elements

The notation presented is for a hidden Markov chain but can be appropriately stripped down and applied to traditional Markov chains. Markov chains can be configured in a variety of ways to emulate specific behaviors. The differences in configuration can be set in the parameters of the Markov chain and translate into the various types of Markov chains that exist today. Various Markov chains, specifically traditional Markov chains, are discussed in the following sections.

There are a few key differences between a Markov chain model and the more complex hidden Markov chain model. In a Markov chain, the model is represented by a number of states and is limited to only the probabilities of transitioning between these states with no observations or relationships to other parameters. In addition, the probability of transition depends only on the current state and no insight into previous events. According to statistical theory, a Markov chain can be said to be memoryless (Ines, 2010). On the contrary, a HMM model is defined by a number of states. However, these states are hidden, where the occupied state remains unknown. Only the observations and outputs that are related to the states detailed above are known. Unlike a traditional Markov chain, an HMM predicts transitions based on previous outputs and not based on the current state. A simple comparison comparing a traditional Markov chain to a hidden Markov chain is provided in Figure 2.2.

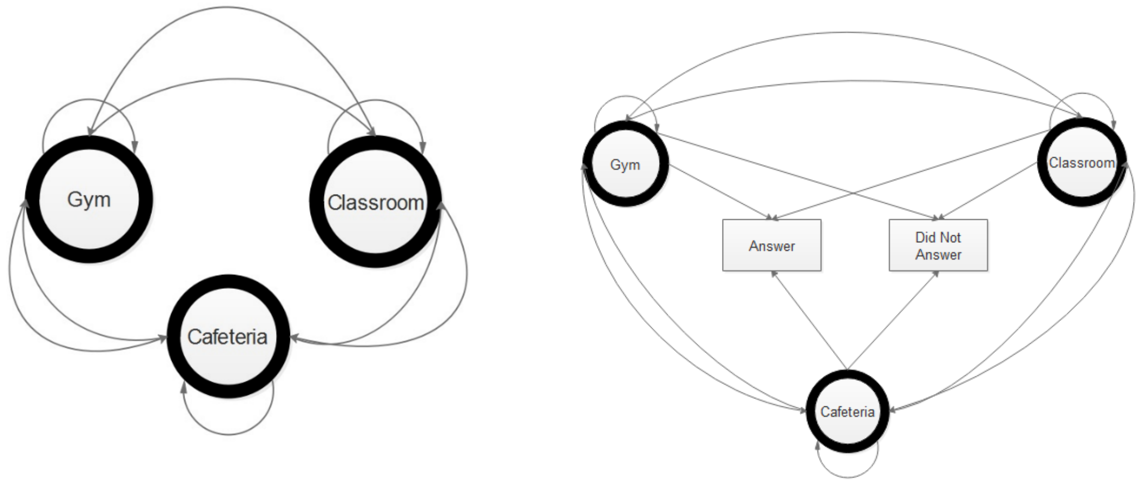


Figure 2.2: Comparison of Traditional (left) versus Hidden Markov Chains (right).

In this example, Student A, who is on a university campus, can be in a classroom, the cafeteria, or the gym as modelled in Figure 2.2 (left). In Figure 2.3 (right), another student, Student B, is attempting to reach the first student by phone. While Student B knows whether or not Student A answers the phone, they do not know whether the student is in class, the cafeteria, or the gym. In Figure 2.2 (right), the classroom, cafeteria, and gym are hidden states, where only the observations (of whether or not the student picks up the phone) are experienced.

Another simple example of an HMM can be illustrated using a weather-related example (Stamp, 2015). The relative temperature of a particular location is in question, and the states and transitions that will occur must be assessed. Temperatures can be either hot or cold (i.e., 2 states), and there exists a likelihood for all four possible transitions (hot to hot, hot to cold, cold to hot, and cold to cold). Suppose a relationship between temperature and plant growth exists, with plant growth ranging from low, to medium, and to and high. The growth, in this instance, represents observations or signals. Depending on the current state (hot or cold), there are associated probabilities that you will observe one of the three growth potentials (i.e., signals). By combining state transition probabilities with observation probabilities, a hidden Markov model begins to be formed.

Since the next state depends on the previous state, this example can be classified as a Markov process. Since we cannot observe the temperature, this model is considered a hidden Markov model. In this case, observable information, or plant growth, can be used to predict temperature. The model is defined based on three key parameters, namely the initial state distribution, the state transition matrix, and the observation matrix. Together, these parameters detail all of the transition and observation probabilities as well as the initial conditions.

These methods, when combined, lend themselves best to long-term projects as there is sufficient time to make necessary, worthwhile changes. For this method to be effective in practice, appropriate data collection practices are a requirement. The mathematical representation of an HMM model is depicted by expressing the binary state vector and/or the probability state vector. These are vectors of length n, where n represents the number of states. The binary vector is an empty zero vector except for the occupied state, which is one. The probability state vector is, again, a vector of length n, where all entries represent the probability of state transition to that state. By definition, the sum of all entries of the probability vector must be equal to one.

2.1.2.3 Ergodic Markov Chain

An ergodic Markov chain is a broad categorization of chains that exhibit the ability to transition to other states from each state in one or more steps. Another name used for this type of chain is an "irreducible Markov chain." A reducibility property refers to the ability to move from a state, i, to a state, j, directly (in one step) or indirectly (in multiple steps). Ergodic Markov chains may be regular (fully connected chain with all transitions having a non-zero probability) or may not be regular (for example, a chain with zero diagonal elements in its transition probability second order tensor).

2.1.2.4 Irregular Markov Chain

In irregular Markov chains, there is absolutely no way of transitioning to all states (including the current state) from the current state within a single step. A classic example is a Markov chain that has a transition probability matrix with zeros along its diagonal (i.e., there are no transitions from a state to itself).

2.1.2.5 Regular Markov Chain

A regular Markov chain is the most frequently encountered type of Markov chain. It is regarded by many as the true ergodic Markov chain. A regular Markov chain guarantees the possibility of moving from one state to any other state (including the current state) in just one step. To be able to fulfill this behavior, the transition probability matrix or its powers must contain elements that have values greater than zero. From a behavioral perspective, this can be viewed as the ability of the Markov chain to transition to every other state.

Regular Markov chains also exhibit other behaviors when certain types of computations are performed on them. For example, there is either one or multiple probability state vectors that, if multiplied by the transition probability matrix, gives the same probability state vector or multiples of the initial vector. In linear algebra, these are referred to as eigen vectors. Eigen vectors have values associated with them that are referred to as eigen values. Together, eigen values and their corresponding vectors form an eigen system. Eigen systems reveal insights into several aspects of the model or process depending on the context in which they are used. For example, in geometric liner transformation of spaces using tensors, eigen vectors represent the portion of the space that does not experience any rotation but that experiences the maximum scaling effect. In essence, the eigen value represents the extent to which the space in the direction of the eigen vectors has been stretched or compressed. In the analysis of the dynamic behavior of physical systems, eigen vectors and values are used to predict the deformation response of structures to dynamic loads, such as earth quakes. The state probability distribution vector of a Markov chain that corresponds to an eigen value of one is referred to as an equilibrium probability distribution of the regular Markov chain. It is also often referred to as the stationary probability distribution. There are a number of analytical techniques for computing the parameters of an eigen system; the process of obtaining these parameters is not of interest here and are, therefore, not discussed.

Also important to note is that applying power operations to a Markov chains transition probability matrix typically ends in a convergence of the result after a certain number of steps. When this happens, the Markov chain is said to have transitioned from a transient state to a state of equilibrium. When the chain has achieved this type of equilibrium, any state probability distribution vector that is multiplied with the transition probability tensor results in the same state probability vector (see Equation 2.1). In other words, the probability distribution of the current step is equal to the probability distribution of all future steps.

$$A^{n} \approx A^{n-1} \tag{2.1}$$

2.1.2.6 Absorbing Markov Chain

Absorbing Markov chains represent a category of Markov models that have more than one absorbing state and have a configuration that permits moving from any state to at least one absorbing state in a finite number of steps (Grinstead & Snell, 1997; Kemeny & Snell, 1976). All non-absorbing states are referred to as transient states. The convention for writing out a transition matrix for an absorbing Markov chain is to number the states so that the transient states come first and then the absorbing states follow (see Equation 2.2). This arrangement conforms to the famous "Canonical Form."

$$P = \begin{array}{c|cc} & \text{TR.} & \text{ABS.} \\ \hline \text{TR.} & Q & R \\ \text{ABS.} & 0 & I \end{array} \tag{2.2}$$

I is an identity matrix that is of dimension r-by-r, O is a zero matrix that is of dimension r-by-t, R is a non-zero t-by-r matrix, while Q is a t-by-t matrix. It has already been mentioned that the element a_{ij}, in a matrix An is the probability of being in the state s_i after n steps, when the chain is started in state s_i. As such, the probability that a Markov chain will be absorbed is 1.0, as the number steps tends to positive infinity irrespective of the state that the chain started in. Absorbing Markov chains are used to model processes that exhibit typical Markov properties but that also terminate after a specific state is realized. The famous drunkard's walk example (see Figure 2.3), where the drunkard's walk is a random walk on a number line where, at each step, the position may change by +1 or −1 with equal probability.

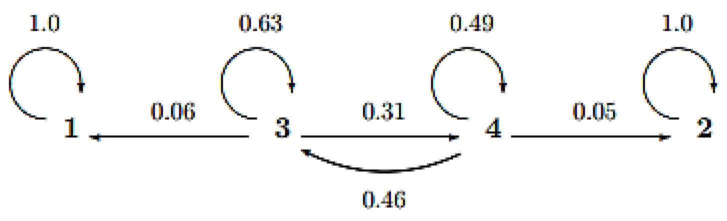


Figure 2.3: A Graphical Representation of a "Drunkard Man's Walk" Markov Chain

2.1.2.7 Markov Chain Behavior

Markov Chains are a classical example of models that are used to analyze and emulate Markov processes. Analysts and modelers have made use of these models by performing simulations or analytical algebraic computations (i.e., often referred to as predictions). Both approaches have been extensively used to gain insights into behavior of Markov processes as time passes.

In the present thesis, Markov process and chain behaviors are defined using two different terminologies to facilitate the discussion of an important point. Prediction of Markov chain behavior refers to the application of liner algebra to the properties of a chain for purposes of obtaining probability and statistical values that can be used for making inferences about the general behavior of a chain. Simulation, on the other hand, strives for precise emulation of the Markov behavior. It also relies on probability theory to achieve randomness that is used to model the actual behavior of the chain. Each of these analyses are discussed further in the following sub-sections.

2.1.2.8 Predicting Markov Chain Behavior

Analysts who are interested in predicting the evolution of process that can be defined using Markov chains often use liner algebra in their computations. For example, an analyst may be interested in estimating the likelihood of being in certain states after a specific number of steps or the expected number of times that a chain transitioned through a specific state in a certain number of steps. In such a case, liner algebra can be applied using the prior state probability vector and the transition matrix of the Markov chain (see Equation 2.3).

$$\pi^{(n)} = \pi^{(0)} A^{n} \text{ ; where:} \tag{2.3}$$

$\pi^{(0)} - Prior\ state\ probability\ row\ vector$
$\pi^{(n)} - State\ probability\ row\ vector\ after\ n\ iterations$
$A - Transition\ probability\ matrix$

The proof of Equation 3 is based on the fact that the state vector for a Markov chain at a specific iteration is the product of the state vector in the previous iteration and the transition matrix (i.e., an application of the Markov or memoryless property). Generating the probability state vector using this principle results in Equation 2.4.

$$\pi^{(1)} = \pi^{(0)} A \tag{2.4}$$

The state probability vector for the second iteration would then be computed using Equation 2.5.

$$\pi^{(2)} = \pi^{(1)} A \tag{2.5}$$

Substituting Equation 5 into Equation 4 results in Equation 2.6.

$$\pi^{(2)} = \left(\pi^{(0)} A\right) A = \pi^{(0)} A^2 \tag{2.6}$$

From Equation 2.6, the equation for computing the state probability vector for the nth iteration can be generalized as Equation 2.7, hence the proof.

$$\pi^{(n)} = \pi^{(0)} A^n \tag{2.7}$$

Using the formulations presented, it can be realized that there is a point at which the probability vector for the chain being in a particular state stabilizes. When this occurs, the chain is said to have achieved equilibrium (Grinstead, 1997). Prior to this, the chain is said to be within a transient state. The number of steps that are required for a chain to achieve equilibrium is often important to Markov analysts. It is possible to use liner algebra to estimate the approximate number of times that a Markov chain, with specific parameters, transitioned through a specific state.

It is also important to note that the results obtained from computations that make use of these equations are estimates and may not represent the actual values obtained when the same chain is simulated. Accordingly, these predictive computations do not represent precise use of Markov chain for the following reasons:

- The results of the predictive computations do not preserve the memoryless property of Markov chains. This is because the elements of the generated Probability State Vectors (PSV) do not match any of the row vectors in the transition probability matrix. Also, the computations rely on the previous powered result of the transition probability matrix—something that is not consistent with the memoryless property of Markov chains.

- As PSV values are computed at each step, they often quickly decay into a set of values that do not fulfill the property that requires all likelihoods of transitions from a state to sum to a value of one. This violates the basic principles of probability, thereby introducing errors.

As a result of these shortfalls, simulation-based approaches are often used as an alternative approach for applying Markov chains to study processes. Details on simulation of Markov chains are presented in the following section.

2.2 Specific Researched Papers of Interest

Probabilistic Simulation Studies for Repetitive Construction Processes (S. AbouRizk, D.W. Halpin) - 1990

The authors demonstrated the use of statistical distributions through input modelling techniques applied to simulation modelling. They illustrate that statistical distributions allow the representation of random inputs of the system being studied. They also detailed two primary cases of input modelling: first, when data is available for fitting, and second, if no data are present. Further detailed is input model selection, model parameter estimation, and goodness of fit assessment.

Simulation Input Updating Using Bayesian Techniques (T. Chung, Y. Mohamed, S. AbouRizk) - 2006

Due to the long-term and repetitive nature of tasks in a construction project, using real-time progress information to adjust simulation models can lead to real improvements in performance prediction. Chung et al. present how Bayesian updating can be applied to enhance the estimation quality of a simulation model using real, observed data. Specifically, the authors describe how normal distributions of parameters within the Simphony simulation environment, such as TBM penetration rate in tunneling, can be updated using Bayes theorem. The contributions of this work allows for the improvement of subjective preliminary assumptions, on which the simulation model is based, to be embellished with actual construction data for more accurate forecasting and prediction.

Chung et al. outlined the value of Bayesian updating techniques applied to tunneling construction to improve the accuracy of predictions, simulation results, and project outcomes. They also discussed the opportunity that tunneling projects present for Bayesian updating improvements due to the long-term, repetitive nature of construction. By detailing the inputs in terms of real project progress, the simulation model could become updated, resulting in the output of more accurate results. These techniques were applied to a case study project in Edmonton, Alberta, to test the validity of the theories. A simulation model, which had been developed during the pre-construction phase, was updated to include actual data to modify the assumptions and metrics in the model such as ground conditions, TBM penetration rates, breakdown information, and other delays using Bayesian algorithms applied to update statistical distributions.

Ground Condition Prediction in Tunneling (H.H Einstein, P.G. Ioannou)- 1987

Einstein and Ioannou (1987) developed techniques to address geological uncertainty in tunneling construction using a discrete-state Markov process. The geological conditions were modelled under a set of geological parameters relating to characteristics focusing the model on rock type. The development of the probability transition parameters was recommended to be either developed through statistical procedures or direct subjective assessment. The statistical procedures depend on the amount and reliability of data available. The estimation of these can be performed using a variety of analytical forms and projections using whatever information is available.

Markovian Geology Prediction Approach and Its Application In Mountain Tunnels (Z. Guan, T. Deng, S. Du, B. Li, Y. Jiang) - 2012

A geology prediction approach, based on Markov random process and Bayesian updating procedure, is presented in this article. It can dynamically and cost-effectively predict future ground conditions in a probabilistic manner as the tunnel face advances and new geological information is revealed. The application case study on the Chuangshi Tunnel project demonstrates that the prediction results were representative of real grounds conditions, especially for the immediate area ahead of the face. This approach can be regarded as a good complement to the geophysical prospecting approach during the construction of mountain tunnels.

The variability of geological parameters was described with respect to their length of persistence and, therefore, were defined as a random process with a state probability being a function of location. Transitions were defined under the product of a transition intensity coefficient (c_i), which relates to condition persistence of state i, and the transition probability (p_{ij}) from state i to j (where i and j represent the current and future state, respectively). The estimation of transition probabilities and intensity coefficients are typically subjective assessment or statistical procedures. If there is sufficient information regarding the geological profile of an area on a project, the coefficients can be estimated by computing the inverse of the average length of each state i, while p_{ij} can be estimated by computing the ratio between the number of transitions from states i to j and the total number of transitions out of state i.

Guan et al. also presented a method to dynamically update the transition intensity matrix obtained from the geology map as new information or observations are available. The following equation, Equation 2.8 was defined to compute these updates:

$$L = [l_{jk}], where\ l_{jk} = P[Y(t_b) = k[X(t_b)] = j] \tag{2.8}$$

Where, $Y(t_b)$ is the observation result at location t_b for the geological parameter $X(t_b)$, and l_{jk} is the likelihood of observation result k given that the parameter state is j.

As real time geological information becomes available, the interval transition probability matrix can be updated sequentially, as demonstrated by Guan et al., in the following manner. Suppose the tunnel face location t_f is excavated towards the direction of a known boring hole location noted as t_q. Considering the interval from location t_f to location t_q., the prior interval transition probability matrix V(t_q-t_f) is calculated by Equation 2.9 using the prior transition intensity A. According to Bayes' theorem, the posterior interval transition probability matrix V_0(t_q-t_f) is updated by this equation, when a certain observation result, k, is revealed at location t_f.

$$V(t1 - t0) = e(t1 - t0)^{*}A \tag{2.9}$$

$V' = [v'_{ij}]$, where $v'_{ij}= (\frac{v_{ij}l_{jk}}{\sum_{j-1}^{n} v_{ij}l_{jk}})$ for a certain observation result, k, at the new location

Then, the posterior transition intensity matrix A', which reflects the "average" transition probability, and the "average" transition intensity coefficient, within the range between location t_q and t_f, can be calculated by Equation 2.10:

$$A' = log\, V' \frac{(t_q - t_f)}{(t_q - t_f)A'} \qquad (2.10)$$

Guan et al. also postulated that ground conditions can often be classified into several grades according to the simultaneous characteristics of several geological parameters of interest. It is suggested that a geology table should be defined to specify which kinds of geological parameter combinations belong to which ground conditions. Considering ground conditions as a complex combination of geological parameters, when the probabilistic characteristic of each single geological parameter is determined by the Markov process presented above, the probabilistic characteristic of a ground condition along the tunnel alignment can be determined.

Geological Prediction and Updating In Tunneling- A Probabilistic Approach (D.B. Ashley, D. Veneziano, H.H. Einstein, M.H. Chan) - 1981

These authors introduced a potential theory to apply prediction and updating techniques to geological conditions. By further defining ground conditions expected to occur, specific construction methods and techniques could be customized for the relevant geological state. They detailed a probabilistic technique to describe geological conditions, which was implemented within a model. With its foundation in Markov-process representation, the method detailed attempts to reduce uncertainty as information is collected. Using only existing data and avoiding more technological and expensive exploration techniques provides an advantage to the construction project through saved time and cost. The technique is based on a discrete-state, continuous parameter Markov model, that is updated using Bayes theorem. The overall procedure was broken down into three main phases, namely preliminary subjective assessment (existing data, and geotechnical reports), secondary explorative data (additional drilling, testing and geotechnical investigation prior to construction), and, lastly, the updating of geological condition information using excavation data during tunneling. The viability of the discussed model was reviewed in a case study of a Power Station discharge tunnel construction project.

Jobsite Logistic Simulation in Mechanized Tunneling (M. Scheffer, T. Rahm, R. Duhme, M. Thewes, M. Konig) - 2014

Scheffer et al. explored logistic processes, supply chain management, and project setup to explain the tendency for mechanized tunneling projects to not meet targeted production rates. A simulation approach, which presents the ability to test and compare various setups of jobsite elements, was developed to analyze the complex system that is mechanized tunneling. The model implemented was based on a process-oriented simulation model created by Rahm et al. (2013), which was embellished to include a logistic model within the AnyLogic commercial software. Special focus was made to establish a connection between external logistics (non-excavation work, production) versus internal logistics or actual tunneling excavation. The use of this model demonstrated the ability to support project management by allowing transparent testing of setups and logistics of the tunneling project and by enabling the provision of further insight into potential logistics-associated project improvements.

Assessing Maintenance Strategies for Cutting Tool Replacements in Mechanized Tunneling Using Process Simulation (A. Conrads, M. Scheffer, H. Mattern, M. Thewes, M. Konig) - 2016

Conrads et al. displayed a performance forecast model for tunneling projects that emphasized the wear and maintenance requirements of TBM cutting tools. They present the notion that, to optimize production rate in tunneling, well-maintained cutting tools are imperative as wear to the TBM face cutting apparatus leads to decreased penetration rates. However, there is a challenge in the ability to monitor the TBM face through a visual inspection without stopping production altogether. If maintenance or repairs are required, there will be even more work stoppage. Applying the empirical tool wear prediction formulae of Köppl et al (2015), the model was built within the AnyLogic software environment using agent-based modelling, discrete-event, and system dynamic simulation techniques. Using a multimethod approach allows the evaluation of maintenance strategies to optimize productivity of the tunneling project. The role of the model was to generate a maintenance management plan capable of optimizing the number of maintenance stops and repairs to the TBM face.

Process Simulation of Mictrotunneling Operations for Productivity Assessment Depending On Ground Conditions (T. Dang, B. Schoesser, M. Thewes) - 2013

This paper details the factors that affect the productivity of microtunneling and attempts to manage the complex interaction of construction processes using process simulation. Using a SysML or System Modeling Language and AnyLogic, a simulation model was developed to create a representative model and assess the efficiency of microtunneling. The theorized model was validated using a real tunneling project for the City of Recklinghausen in Germany. A secondary study was completed to assess various soil compositions and their impact on process efficiency. Further sensitivity analysis was done to gauge various combinations of resources, highlighting the most notable processes that affect productivity.

Uncertainty Modeling and Simulation of Tool Wear in Mechanized Tunneling (T. Rahm, R. Duhme, K. Sadri, M. Thewes, M. Konig) - 2013

This paper postulates that geotechnical constraint uncertainty can be estimated using Fuzzy Logic techniques to predict the relationship between geological condition, cutting tool wear, and, consequently, TBM advance rate. Simulation modelling was done to quantify the influence of tool wear on advance rates and to move towards further defining a holistic simulation model of mechanized tunneling projects. The results of the study demonstrated that in the artificial example used, there is significant correlation between advancement of the TBM and wear of cutting tools. Further recommended studies are pursued, and modelling is applied to a real-life case study.

Integrated Pavement Management System with a Markovian Prediction Model (K.A. Abaza, S.A. Ashur, I.A. Al-Khatib) - 2004

Abaza presented a pavement management system that utilizes a discrete-time Markovian model to predict the deterioration to improve scheduling of maintenance and rehabilitation works. Using five condition states, the pavement status can be predicted with the use of system modules built with the Fortran language. Abaza emphasizes that the success of the system is highly dependent on the level of input data and modelling available.

Simulation Case Study: Modelling Distinct Breakdown Events for a Tunnel Boring Machine Excavation (M. Werner, S. AbouRizk) - 2015

Tunnel Boring Machine (TBM) tunneling projects frequently experience delays, which can cause adverse effects, such as extended schedules, which incur additional costs. This paper outlines a case study to demonstrate how simulation can be effectively used to analyze the productivity performance of a project, with an emphasis on delays resulting from equipment breakdowns and unexpected conditions. Data collected from this project under a Method Productivity Delay Modelling study, completed by a consulting firm, was collected and prepared to model delays on a combined discrete-event, continuous tunneling simulation model. Calibration was done to the theoretical tunneling model to ensure the results would be reflective of the actual construction project and to measure the effectiveness of the delay modelling. Sensitivity analysis was conducted to determine the most unfavourable delays for a tunneling project, allowing further analysis regarding the results of the mitigation of these delays on project duration and on hypothetical costs.

Planning the Handling of Tunnel Excavation Material – A Process of Decision Making Under Uncertainty (Ritter, S., Einstein, H., & Galler, R.) - 2013

Previous work in predicting geological conditions in tunneling simulation has been performed using a technique known as Preliminary Material Classification (Ritter et al., 2013), whereby each excavated material is related to a corresponding material class. The preliminary ground class profile is determined based on geological parameters such as lithology, water occurrence, and overburden. A Monte Carlo procedure is used to simulate the construction process, where one of the many defined ground class profiles is computed. As the tunnel progresses, each of the ground profiles is simulated using a construction module that defines the tunneling parameters based on the given ground profile (Ritter et al., 2013).

Schedule Risk Analysis for TBM Tunneling Based on Adaptive CYCLONE Simulation in a Geologic Uncertainty-Aware Context (Donghai, L., Peng, X., Shuai, L., & Peizhi, H.) - 2015

A geological prediction model summarized by Donghai Liu (2015) and developed by Ioannou (1987), used Bayesian techniques based on a Markov process. This model relies upon expert interviews to gauge the accuracy of the transition probabilities of the prediction model. The model functions by predicting future states, based on the previous states, and updating the transition probabilities accordingly.

2.3 Forecasting of project performance

The ability to control and understand performance variances and to predict future project performance is vital for project success. Forecasts of the completion variances are typically estimated by comparing the baseline values with forecasted values. The ability to forecast performance during the project would allow project managers to implement change and take action to mitigate issues where applicable. It is important to note that, on a monthly basis, forecasting is consistent, analytical, and stable so that performance overruns can be identified as early as possible. This document details the work of Nadim Nasser (2005), which focuses on the forecasting of project performance from up-to-date performance data using Markov-based techniques.

Accounting for certain factors during a construction project, such as material delays, scope changes, productivities, and weather, can make performance forecasting difficult to perfect. Typical forecasting methods do not address these variables and do not allow users to input that could improve forecasting. Being able to predict project performance at completion is imperative for ensuring project success and for maximizing profitability. With effective forecasting techniques in place, the management team can address variances and take corrective action as early as possible to remedy overruns. Ultimately, a complete and integrated forecasting method that considers the probabilistic nature of construction projects is essential for allowing management to actively address variances at any future time and, most importantly, at completion.

In the industry today, forecasting is typically done using indices that are typically based on expert opinion and/or linear trends. Markov Chains have commonly been used in many other domains to address stochastic processes; however, this practice is not common in construction performance forecasting. The work of Nasser introduces forecasting theories using objectives hierarchy and Dynamic Markov Chains.

2.3.1 Other Forecasting Methods

Significant work has been done to advance performance prediction. The primary focus of these techniques relate to the prediction of estimates at completion relating to cost and schedule. Various types of techniques have been used, including, but not limited to, stochastic and probabilistic methods, advanced computing methods, earned value methods, deterministic methods, artificial intelligence and fuzzy logic, and behavioral and judgmental forecasting.

2.3.2 Probabilistic Forecasting of Project Performance using Markov Chains

In general, project performance is a function of current status, future corrective action plans, and the incorporation of past performance. The following section details the method formulated by Nasser in his work entitled “An Integrated Framework for Evaluation, Forecasting and Optimization of Performance of Construction Projects (2005)”, which serves to predict future construction performance (at completion and at any interim period) based on a user’s previous experience and judgement. Markov chains were proposed to account for the stochastic nature of construction and performance.

Nasser indicated that application of Markov chains to construction are common and have been applied across a variety of contexts, including assessing the performance of highway bridges, operations of a tunnel-boring machine (Touran, 1997), optimization of maintenance of infrastructure networks (Guignier and Madanat, 1999), structural deterioration of storm water pipes (Micevski, 2002), performance of bridge paint systems (Zayed, 2002a), life cycle economic analysis by (2002b), deterioration model for concrete bridge decks (Morcous, 2003), and deterioration of pavement systems (Abaza, 2004).

CHAPTER 3

[1]PROPOSED GEOLOGICAL CONDITIONS UPDATING USING BAYES THEOREM AND MARKOV CHAIN

3.1 Introduction

A major source of uncertainty in tunneling construction is geological uncertainty, which, if improperly managed, can lead to additional costs and project delays. Tunneling projects are often based on weak or incomplete assumptions, renderings the planning such projects difficult in practice (Ioannou, 1987). Increasing the certainty with which ground conditions are predicted would allow practitioners to select construction method(s) and develop construction plans that are best suited for the soil type.

However, a complete, and actual profile of the ground conditions, which can be obtained through geological borehole investigation, is an expensive, time-consuming process that is often unfeasible in practice. Alternatively, uncertainty can be reduced by applying previous project data or expert knowledge to estimate the unknown parameters of a project. For example, data from a previous project that occurred in close proximity to the new project can be used to inform future construction plans. Furthermore, inclusion of data from the current project as it is acquired can also be used to reduce project uncertainty (Einstein, 2004). Yet, the following question remains: what is the best method to make use of previous and current project data to embellish project plans?

Based on conditional probabilities, Bayes updating techniques provide a systematic approach for combining observed and historical data together to expand existing data sets. Bayes techniques allow the modelling of changes over time and can serve to effectively model the evolution of the probabilistic dependencies within a system. Bayes updating can considerably improve the quality of input data, even when only a small number of data sets have been collected in the early stages of a project's lifecycle. A modeller is then able to easily update predictions as additional information or data becomes available (Špačková & Straub, 2013; Zhang et al., 2014).

[1]A version of this chapter has been published as "Proposed Geological Conditions Updating Using Bayes Theorem And Markov Chain" in the *Proceedings of the 2015 Winter Simulation Conference*.

Modelling the real world, in terms of predictions, is a stochastic process, since the process begins in one state and transitions to another state in a random manner. To address the stochastic nature in real world applications, Markovian chains, named after Andrey Markov, are often used to represent this process. (Gilks, 2005).

This chapter develops a model consisting of Bayes technique and Markov chain, where by the geological condition of a one-meter section in a tunnel construction project can be updated in a continuous manner from observations of the previous one-meter section over time. A case study project in Edmonton, Alberta, the West Edmonton Sanitary Sewer (WESS), was used to demonstrate the applicability of the developed methodology. Results of analysis, including observations, geological fluctuations, as well as additional geological investigation, are analyzed and discussed. The use of the discussed updating techniques demonstrates that prediction quality of the model can be considerably improved following the implementation of the proposed methodology.

3.2 Objective of the Study

The tunnel boring machine (TBM) advancement rate is stochastic and depends on the type of soil or ground condition being excavated (AbouRizk & Mohamed, 2000), which, in turn, impacts the productivity level of the excavation and the project. Typically, in previous tunnel simulation research, the distribution of ground conditions is fixed to certain ranges along the alignment. For example, for a 600m tunnel, it may be assumed that the first half of the tunnel is sand and, after 300m, the remainder of the soil is clay (Ruwanpura et al., 2001). However, geological conditions can change at any time and often do not follow these predefined ranges (Einstein et al, 1999; Haas & Einstein, 2002).

The objective of this chapter was to formally represent the uncertainty of ground conditions for a tunneling project to more reliably predict the actual distribution of geological conditions. Specific objectives of this research included:

1) Modeling the random evolution of the probabilistic dependencies on the distribution of geological conditions
2) Updating the prediction of geological conditions over time when the observed information is available.

3.3 Methodology

By definition, an updating process functions uses new information to improve or update a prior state or set of information. In the case of a Markov model, the goal of an update is to improve predictions. Each probabilistic input is a prediction of an individual parameter that is attempted to be improved during the updating process. As a result, this will also improve the output of the model, leading to a reduction in the level of uncertainty. A proposed framework was developed as shown in Figure 3.1, to visualize the Markov updating process. This framework consists of two steps:

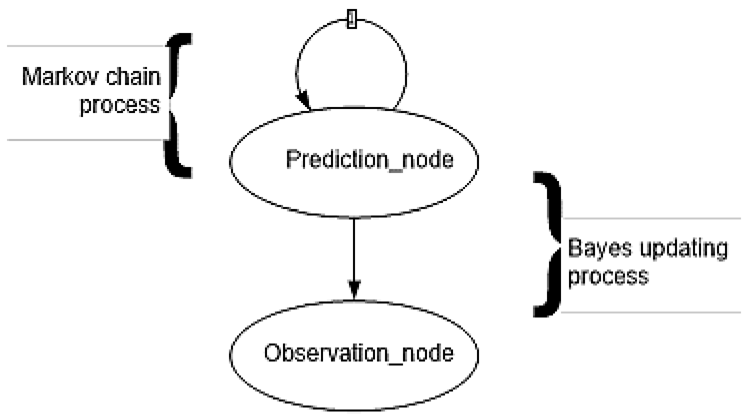


Figure 3.1: A General Simulation Framework

3.3.1 Bayes Updating Process

Under the assumption that a prediction has already been made and that new information has become available since the initial prediction was made, the updating process then leads to an updated prediction. The proposed mathematical model that completes the updating of the probabilistic inputs is Bayes theorem. Bayes theorem is based on the Bayes conditional probability theorem, which is well known in the fields of mathematics and statistics. This theorem allows the modelling of the changes in the probabilistic values within a random system and, in the case of Markov modelling, can allow users to easily update the predictions as new information is known. In other words, Bayes theorem states that the posterior prediction is a function of the prior prediction and the new information.

In a tunneling project, new information can be obtained by performing preliminary testing (borehole sampling), through excavation, or through exploratory testing during excavation. Bayes theorem relates the posterior probability to the prior probability and new information, as shown in Equation 3.1.

$$P''(Z = z_i|X = x_i) = N * P'(Z = z_i) * P(X = x_i|Z = z_i) \tag{3.1}$$

Where, $P''(Z = z_i|X = x_i)$ represents the posterior probability z_i, given that the outcome of the test X is x_i; $P'(Z = z_i)$ represents the prior probability of z_i; and $P(X = x_i|Z = z_i)$ represents the probability of the outcome being x_i if the true state of Z is zi.

3.3.2 Markov Chain Process

The Markov process is a stochastic process that involves both a random variable and a "time" parameter that monotonically increases during the process. The latter representing the state transitions of the random variable as time progresses. The Markov process is characterized by a single-step memory, where only the current state is known. To determine the next step, all previous steps, except for the most recent one, are neglected. When applied to tunneling, this parameter indicates that the material behind the tunnel face depends only on the parameter state at the tunnel face.

A discrete-time Markov chain is a mathematical system that simulates transitions from one state to another in the real-world. Based on the Markov chain process defined in Equation 3.2, the model is a probability distribution function of the sequence of T hidden-state variables X = {x_0, . . ., xT-1} and the sequence of T observations Y = {y_0, . . ., yT-1} that has the following factorization, which satisfies the requirements that state x_t depends only on state x_{t-1}.

$$\Pr(X_{1:\,T}|Y_{1:\,T}) = \prod_{t=1}^{T-1} \Pr(x_t|x_{t-1}) \times \prod_{t=0}^{T-1} \Pr(y_t|x_t) \times \Pr(x_0) \tag{3.2}$$

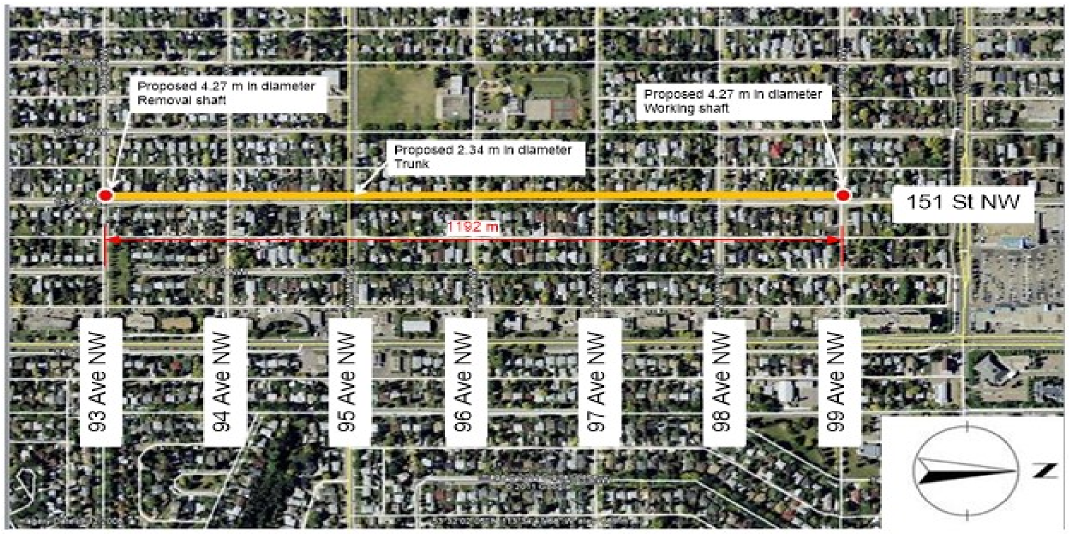
Where, $P_r(x_t|x_{t-1})$ is the state transition probability density function, $P_r(y_t|x_t)$ is the observation probability density function, and $P_r(x_0)$ is the initial state distribution.

3.4 Case Study

A tunnel case was used to assess the feasibility of the proposed Markovian model and, specifically, Bayesian updating. In tunneling construction, concrete liner is usually installed in one-meter lengths. To conveniently analyze the tunnel project and its production rate, this model divides the tunnel into one-meter sections, which are considered as separate units during excavation.

3.4.1 Background

The WESS was a sanitary servicing project, performed in anticipation of future growth, that initiated west of the City of Edmonton to prevent negative impacts on the downstream sewer system. This project consisted of approximately 1.2 kilometers of underground tunnel construction (Figure 3.2).

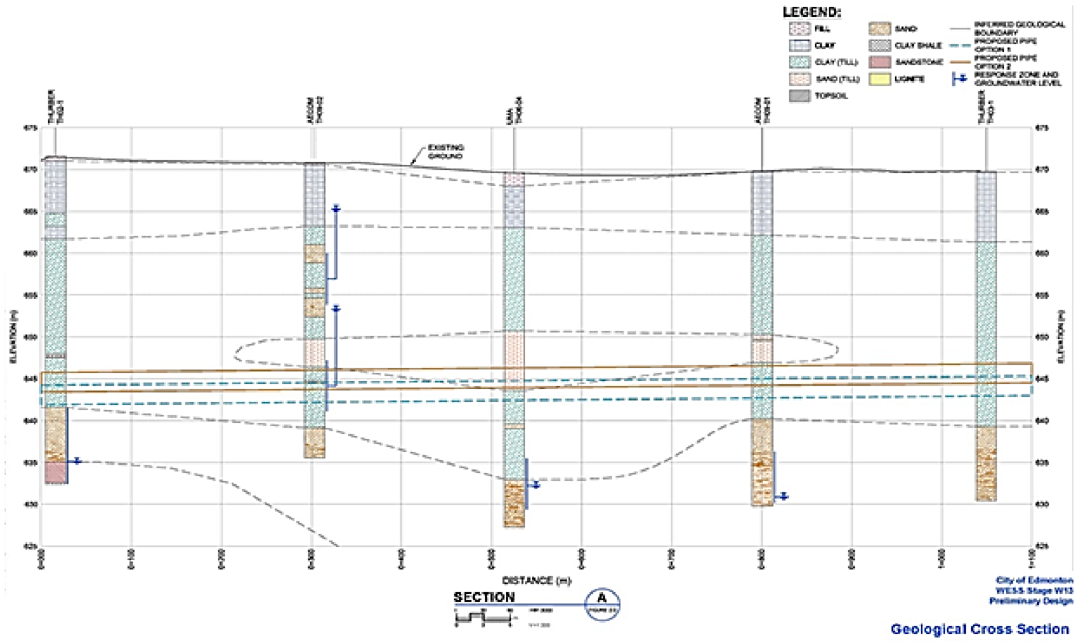


[2]**Figure 3.2: Location of the WESS W13 Project**

The production of tunnel construction projects is very sensitive to geological conditions due to complex tunnel-soil interactions. Certain ground conditions, such as very loose sand, may cause hazardous tunnel operations or catastrophic failures of a tunnel boring machine (TBM). To determine the underground geological profile of the WESS project, six preliminary boreholes were drilled to depths of approximately 35 and 40 meters below ground surface (mBGS). Standard

[2]Figure provided by SMA Consulting Ltd. as part of geological investigation reports.

penetration tests (SPT) were performed at regular intervals in the boreholes and SPT soil samples were collected and visually classified in the field according to the Modified Unified Classification System for soils. Figure 3.3 illustrates the result of the geological investigation of each cross section for the WESS W13 project.



[3]Figure 3.3: Geological Investigation of Cross Section for the WESS W13 Project

3.4.2 Model Construction

As previously discussed, ground conditions are a major variable that dictates excavation and, consequently, TBM productivity. Knowledge of ground conditions are important for planning and optimization of tunneling projects, as specific TBM characteristics, such as specific cutting teeth, can be tailored to certain ground conditions.

In engineering and construction, geotechnical engineers will commonly classify soils or geological conditions based on their engineering properties pertaining to their structural properties for use as foundational support or as a component of building material. In North America, standardized soil classification typically follows the Unified Soil Classification System (USCS). The USCS defines three major classifications: highly organic soil (e.g., peat), fine-grained soils (e.g., silts and clays)

[3]Figure provided by SMA Consulting Ltd. as part of geological investigation reports.

or coarse-grained soils (e.g., sands and gravels). The following analysis in this chapter follows the proposed classification used by Einstein et al., (1999), which use three classifications, namely "little," "medium," and "intense," which represent the three USCS groups noted above, respectively.

The objective of the following work is to update the geological conditions using new observations over time. To determine the geological condition in a detailed and precise manner, the entire tunnel length is divided into shorter cross sections, such as one-meter sections, that are almost equal to the width of a segment (see Figure 3.4).

[4]**Figure 3.4: Segments for the WESS W13 Project: (a) Segments to the Site and (B) Segments in the Tunnel**

We assume that each one-meter section only has one specific soil type, rather than a combination of various soil types. For simplicity, the construction layout of the cross section is abstracted into a rectangle with 1192 one-meter sections, as shown in Figure 3.5.

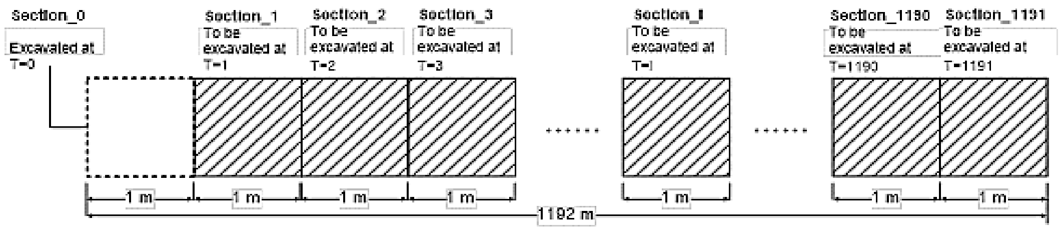


Figure 3.5: Layout of 1192, Nine-Meter Sections on the Construction Site for the WESS W13 Project

[4]Figure provided by SMA Consulting Ltd. as part of geological investigation reports.

The dotted box shown in left part of Figure 3.5 represents the section that has been excavated at T=0. The solid boxes as shown in the right part of Figure 3.5 represent the sections that will be excavated at T=1~1191, respectively. The observed geologic conditions at the previous time slice provide new information that is used to update the geologic conditions of the subsequent slice.

To analytically make use of the interactions between neighboring one-meter sections throughout the continuous excavation process (i.e., transitions between soil conditions), the updating chain for the prediction of soil conditions, based on the Markov chain and Bayes theorem, is established and shown in Figure 3.6. The established model consists of a series of sub-models (see Figure 3.1). The virtual arrow lines that connect neighboring sub-models indicate that the posterior probability distribution of the ith section at $T=t_i$ can provide soft evidence information for the observation of the ith section at $T=t_{i-1}$ in cases where no observation information is available. With regard to the ith specific submodel ($1 \leq i \leq 1091$), two nodes that are involved in the updating process, predition_i and observation_i, exist. Specifically, the node of prediction_i stores the predicted probability distribution, while observation_i stores the observed information from the previous time point. In this study, since the geologic conditions are classified into three types, each node, therefore, has three states.

According to the geological condition profile for the WESS W13 project (see Figure 3.3), the tunnel alignment is mainly located in a layer of clay-till and sand. The surrounding soil tends to be classified as "medium" throughout the entire cross section, albeit with slight fluctuations. Thus, the conditional and transition probability distributions between Bayes nodes remains consistent in each submodel. Table 3.1 illustrates the conditional probability table (CPT) between prediction_i and observation_i. Table 3.2 illustrates the transition probability table (TPT) of prediction_i. Table 3.3 illustrates the prior probability distribution of prediction_i at time T=i.

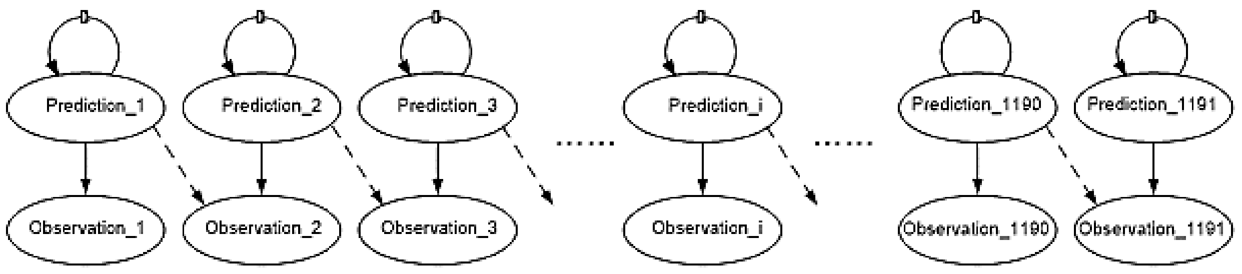


Figure 3.6: Updating Chain for the Prediction of Soil Conditions

Table 3.1: Conditional probability table (CPT) between prediction_i and observation_i.

Observation_i states	**Predition_i states**		
	Little	**Medium**	**Intense**
Little	0.7	0.15	0.1
Medium	0.2	0.7	0.2
Intense	0.1	0.15	0.7

Table 3.2: Transition probability table (TPT) of prediction_i.

Prediction_i states (T=t)	**Predition_i states (T=t-1)**		
	Little	**Medium**	**Intense**
Little	0.8	0.1	0.05
Medium	0.15	0.8	0.15
Intense	0.05	0.1	0.8

Table 3.3: Prior probability distribution of prediction_i.

Prediction_i states	**Prior Probability**
Little	0.1
Medium	0.8
Intense	0.1

As shown in Figure 3.3, the geologic conditions are investigated by means of a few distributed boreholes and, therefore, the geologic conditions are not known prior to excavation. Using the proposed methodology, real-time information collection throughout the excavation process can be used to update the posterior probability distribution for future soil time slice predictions. The direct result of this analysis is that engineers or construction personnel can apply this methodology to reduce the uncertainty that is traditionally associated with soil conditions ahead of the current excavation face. To verify the applicability of the proposed approach, the impacts of given observations, geological fluctuations, as well as additional geological investigation, were tested for 12 different scenarios.

3.4.3 Results

Observations at the current time slice should have an impact on the posterior probability distribution of the predicted node. To test the degree of impact, an experiment with four different scenarios was conducted. Results of the posterior probability distribution of the predicted node over time for various observations at T=0 are presented in Figure 3.7. For Scenario 1, no given observations are provided at T=0; in Scenario 2, the type of “little” is observed for the section_0 at T=0; in Scenario 3, the type of “medium” is observed for the section_0 at T=0; and in Scenario 4, the type of “intense” is observed for the section_0 at T=0. The first 30 one-meter sections (see Figure 3.5) were chosen to demonstrate the impact of various observations on the posterior probability distribution.

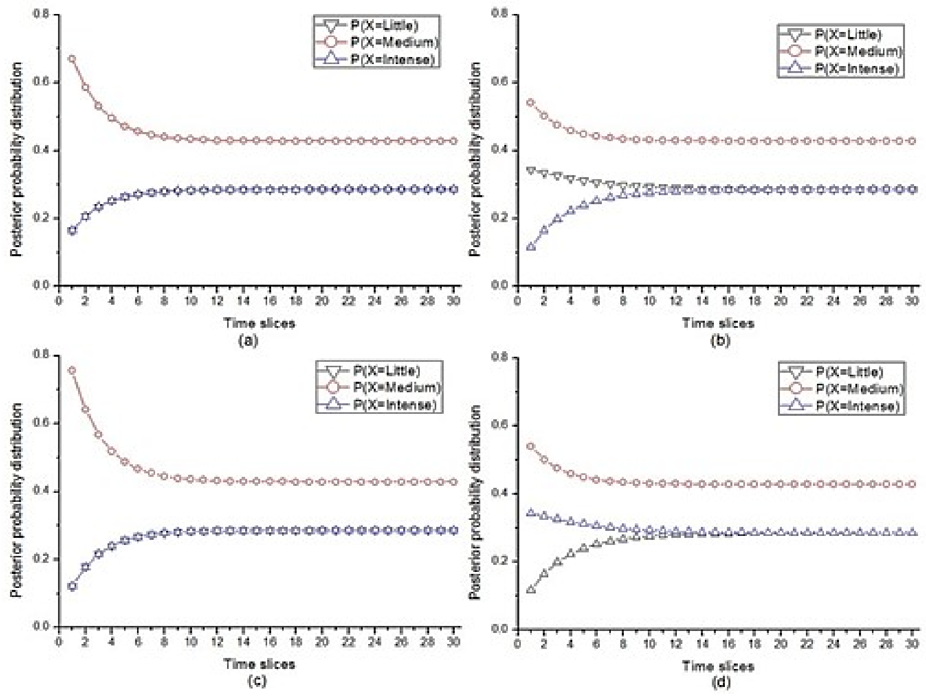


Figure 3.7: Results of the posterior probability distribution of the predicted node over time in case of different observations at T=0: (a) Scenario 1 (no given observations are provided at T=0); (b) Scenario 2 (the type of “little” is observed for the section_0 at T=0); (c) Scenario 3 (the type of “medium” is observed for the section_0 at T=0); and (d) Scenario 4 (the type of “intense” is observed for the section_0 at T=0).

The results indicate that the posterior probabilities vary distinctly depending on the observations at T=0. The posterior probability distribution is very sensitive to the observed observations, indicating the established model is valid to some extent. Alternatively, the impact of given observations at one specific time segment drops gradually in the subsequent time segments, and tends to be very modest after approximately ten-time segments. This updating technique is only a brief window into the overall picture, and the range is limited. Therefore, a series of continuous observations are very meaningful for reducing future underlying uncertainties.

As previously shown in Figure 3.3, the tunnel for the WESS W13 is mainly located in a layer of clay-till and sand. Thus, the prior probability regarding the type of “medium” is set to 0.8, and those of “little” and “intense” are set to 0.1 for all prediction nodes in the updating chain (see Table 3.4). However, geological fluctuations are common in construction practice due to uncertainties underlying complex underground environments. To analyze the impact of geological fluctuations on the posterior probability distribution of the predicted node over time, an experiment with two different scenarios is detailed below. Figure 3.8 illustrates the evolution of the posterior probability distribution of the predicted node over time in Scenarios 5 and 6. In Scenario 5, the type of “medium” is observed at T=0~5, and the type of “little” is observed at T=6~8; in Scenario 6, the type of “medium” is observed at T=0~5, and the type of “intense” is observed at T=6~8. The first 30 one-meter sections (see Figure 3.7) are chosen to model the impact of geological fluctuations on probability updating.

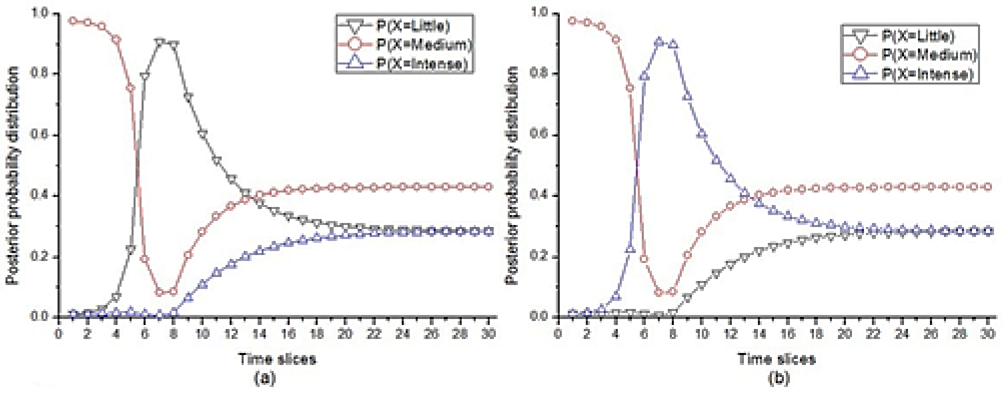


Figure 3.8: Results of the posterior probability distribution of the predicted node over time in case of a geological fluctuation: (a) Scenario 5 (the type of “medium” is observed at T=0~5, and the type of “little” is observed at T=6~8); and (b) Scenario 6 (the type of “medium” is observed at T=0~5, and the type of “intense” is observed at T=6~8).

The results in Figure 3.8 indicate that the observation of the geological fluctuation can lead to a sudden fluctuation of the posterior probability distribution. In Scenario 5, the probability for the type of “little” experiences a spike and subsequent fall through T=6~12 based on a geological fluctuation from the type of “medium” to “little.” In Scenario 6, the type of “intense” experiences a spike and fall at T=6~12 in case of a geological fluctuation from the type of “medium” to “intense.” In both scenarios, it is notable that the influence of the geological fluctuations on probability are seen to be extended to the subsequent 16 time slices. Effects beyond this point appear modest.

During actual tunnel construction, when an undiscovered geological condition is observed, such as a sudden geological fluctuation, additional geological investigations are often performed to estimate the length of the newly discovered condition. To analyze the impact of such additional geological investigations on the posterior probability distribution of the predicted node over time, an experiment with six additional scenarios was performed. Figure 3.9 illustrates results of the posterior probability distribution of the predicted node over time in Scenarios 7 to 12. Here, in Scenario 7, the type of “medium” is observed at T=0~5, the type of “little” is observed at T=6~8, and the type of “little” is observed at T=18~20; in Scenario 8, the type of “medium” is observed at T=0~5, the type of “little” is observed at T=6~8, and the type of “medium” is observed at T=18~20; in Scenario 9, the type of “medium” is observed at T=0~5, the type of “little” is observed at T=6~8, and the type of “intense” is observed at T=18~20; in Scenario 10, the type of “medium” is observed at T=0~5, the type of “intense” is observed at T=6~8, and the type of “little” is observed at T=18~20; in Scenario 11, the type of “medium” is observed at T=0~5, the type of “intense” is observed at T=6~8, and the type of “medium” is observed at T=18~20; and in Scenario 12, the type of “medium” is observed at T=0~5, the type of “intense” is observed at T=6~8, and the type of “intense” is observed at T=18~20. The first 30 one-meter sections are chosen to demonstrate the impact of additional geological investigations on probability updating.

As shown in Figure 3.9, the results indicate that the observations obtained from additional geological investigations can provide new observations for probability updating from the zone of the currently excavated section to the additionally investigated section, as well as the zone ahead of the additionally investigated section. Specifically, in the zone from the currently excavated section to the additionally investigated section, a “U” type shape (see Figure 3.9 (a) and (f)) develops where the geological condition of the above sections remains consistent, and an “X” shape of “X” (see Figure 3.9 (b) ~ (e)). In the region ahead of the sub investigated section, the posterior probability of the observed soil type in the additionally investigated section drops gradually in the subsequent time slices, and again, it is shown that the influence desensitizes over time.

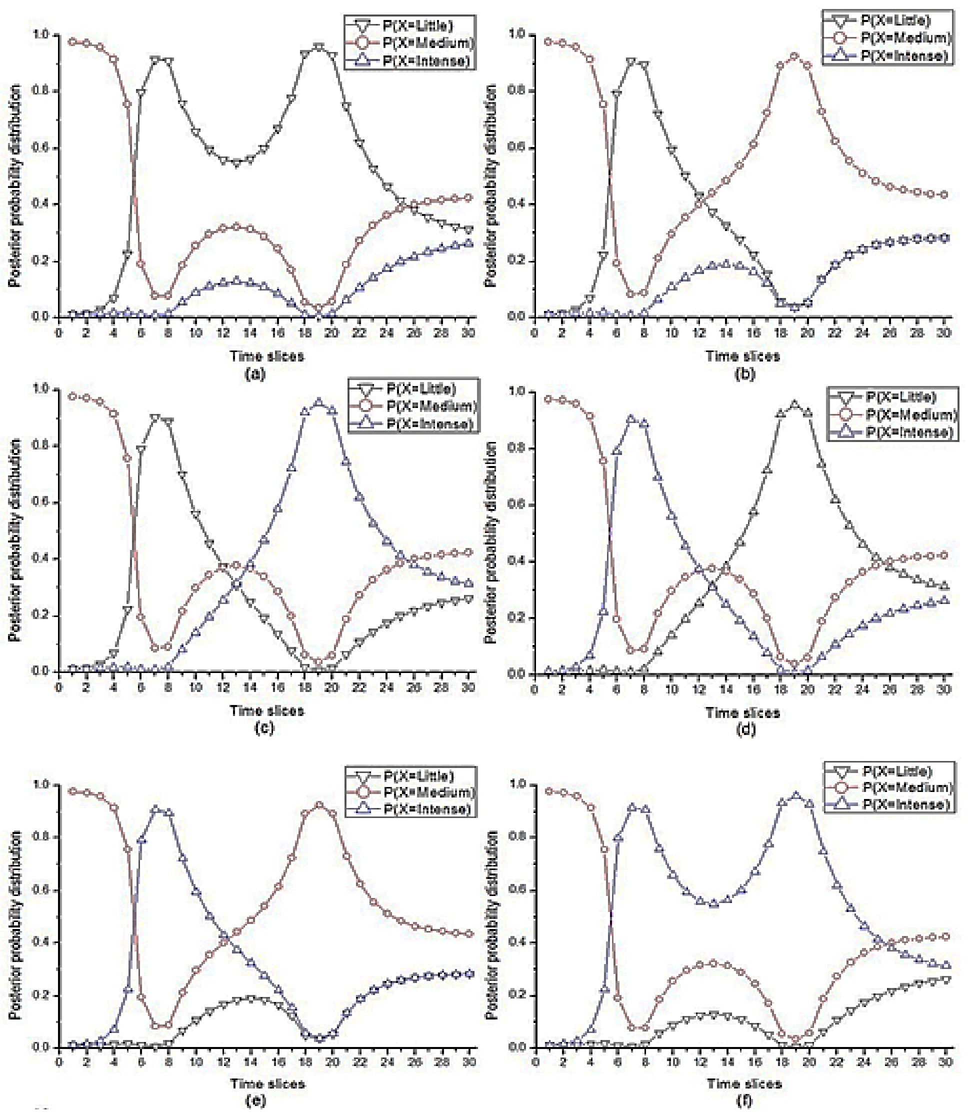


Figure 3.9: Results of the posterior probability distribution of the predicted node over time in the case of additional geological investigations: (a) Scenario 7 (the type of "medium" is observed at T=0~5, the type of "little" is observed at T=6~8, and the type of "little" is observed at T=18~20); (b) Scenario 8 (the type of "medium" is observed at T=0~5, the type of "little" is observed at T=6~8, and the type of "medium" is observed at T=18~20); (c) Scenario 9 (the type of "medium" is observed at T=0~5, the type of "little" is observed at T=6~8, and the type of "intense" is observed at T=18~20); (d) Scenario 10 (the type of "medium" is observed at T=0~5, the type of "intense" is observed at T=6~8, and the type of "little" is observed at T=18~20); (e) Scenario 11 (the type of "medium" is observed at T=0~5, the type of "intense" is observed at T=6~8, and the type of "medium" is observed at T=18~20); and (f) Scenario 12 (the type of "medium" is observed at T=0~5, the type of "intense" is observed at T=6~8, and the type of "intense" is observed at T=18~20).

3.5 Conclusion

The geological prediction and updating model detailed presents a means for explicitly making use of new information as well as data customarily used by engineers and geologists to assess the geologic conditions of a tunneling project. The developed model, consisting of Bayes technique and Markovian modeling, is able to incorporate newly acquired information to model the uncertainty around geological conditions over time. It was found that the prediction of geological conditions of one-meter sections in tunnel construction can be updated in a continuous manner from the geological condition of the previous one-meter section that is observed as the construction progresses. A real project of the WESS was used to verify the applicability of the developed model. The model test results demonstrate that continuously updating data sets during construction can significantly improve the prediction of the project performance by eliminating the uncertainty contained in the original assumption. The models updating procedure can be easily expanded to predict the results of future geological exploration programs, as well as project productivity and quality, as a function of the reliability of the observation methods employed. The model discussed presents the potential for integration of Markovian and Bayesian techniques and provides advocacy for additional research and work in the subject are.

CHAPTER FOUR

PROTOTYPING AN HMM BAYESIAN UPDATING ALGORITHM FOR DEPLOYMENT IN A SIMULATION ENVIRONMENT

4.1 Background

Markov models are the most appropriate method to use for situations that require the analysis of processes that can be represented by sequential data. A hidden Markov model (HMM) is a special type of Markov model that assumes that the states of the system or process being modelled are not visible to the analyst. Details of these states are assumed and modelled based on observable sequences. There are a number of assumptions that are carried over from the traditional Markov models to HMMs, including the assumption that (1) the next state (i.e. the i^{th} hidden state) is dependent on only the previous state (i.e. the $(i-1)^{th}$ hidden state) and (2) the current observation variable is only dependent on the current hidden state.

The set of observations in the case of HMMs is a sequence of various observable outcomes. Unlike other Bayesian updating methods, both the unique instances in the data sequence and the transition from one outcome to the next, impact HMM updating. The transition-associated information and specific instances serve as inputs for the updating step. Training of HMMs is not as straightforward as that for statistical distributions. Consequently, analytic methods cannot be applied for this purpose, and numeric methods are a more suitable option for such situations (Rabiner, 1989). Several numerical methods have been proposed for updating the parameters of HMMs, such as Genetic Algorithms (Manabe et al., 2006), Particle Swarm Optimization (Novak & Macas, 2008), and Baum-Welch (Rabiner, 1989). This chapter discusses the Baum-Welch algorithm for the purpose of training HMMs. The Baum-Welch method was selected for implementation in this version of the prototype because it has the most transparent process or steps compared to the other methods.

4.2 Baum-Welch Algorithm for Training HMMS

According to Rabiner (1989), determining a method to adjust HMM parameters (i.e. A, B, π) to maximize the probability of the observation sequence from a given model is difficult, as there is no way to analytically solve this problem. Rather, Rabiner (1989) has suggested the use of iterative numeric methods, such as the expectation modification method (Dempster et al., 1977) and gradient techniques (Levinson et al., 1983).

Here, the Baum-Welch is the specific EM method adopted. This algorithm is numeric in nature and was first proposed by Leonard E. Baum and Lloyd R. Welch in the late 1960s (Baum & Petrie, 1966; Baum & Eagon, 1967; Baum & Sell, 1968; Baum et al., 1970; Baum, 1972). Their work was based on the forward-backward procedure described by Rusian Stratonovich (1960).

The Baum-Welch algorithm involves utilizing a set of observed outcomes to update the parameters of an HMM. These observed outcomes may be a simple set containing one array of observations or may be complex with a series of observation arrays.

In the development work presented in this chapter, it is assumed that the algorithm will take the case of a simple observation set as its input. However, there is a 'work around' with this algorithm that a modeller may deploy to deal observation complexity, which involves using each array of observations as a simple set and updating the HMM accordingly. In the first case, the original HMM is updated. This updated HMM is then utilized as the original HMM in the next array of observations. This is repeated until all the arrays are utilized. The updated HMM from the last array of observations is then taken as the final output.

Figure 4.1 summarizes the topology of the different types of observation sets likely to be encountered in HMM updating situations.

$$\textit{Simple set of observations} = \{x_1, x_2, \ldots\ldots\ldots, x_n\}$$

$$\textit{Complex set of observations} = \{s_1, s_2, \ldots\ldots\ldots, s_n\}$$

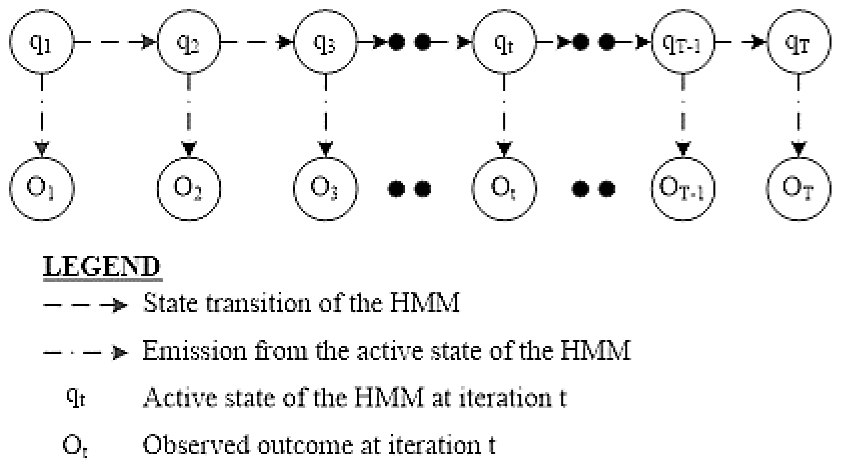
$$\begin{aligned}
&\textit{Where:}\\
&s_1 = \{x_{11}, x_{12}, \ldots\ldots, x_{1m}\}\\
&s_2 = \{x_{21}, x_{22}, \ldots\ldots, x_{2m}\}\\
&- \quad - \quad - \quad - \quad - \quad - \quad -\\
&s_n = \{x_{n1}, x_{n2}, \ldots\ldots, x_{nm}\}
\end{aligned}$$

Figure 4.1: Type of Observable Data Sequences

The algorithm utilizes the set of observations to generate parameters that, in this chapter, are referred to as updating parameters. These updating parameters represent probability values and are used to update the parameters of the HMM. The updating parameters output by the algorithm include: (1) alpha values (α), (2) beta values (β), (3) zeta values (ξ), and (4) gamma values (Υ). Gamma and zeta updating parameters are computed based on alpha and beta parameters. These two parameters (i.e., gamma and zeta) are the only ones used in the final updating step.

4.2.1 Computing Updating Parameters Using Baum-Welch Algorithm

To simplify the interpretation of the updating parameters, a Trellis diagram is presented (See Figure 4.2). This diagram illustrates the states that the HMM transitions through as observations are being made.



q_1
q_2
q_3
q_t
q_{T-1}
q_T
O_1
O_2
O_3
O_t
O_{T-1}
O_T
LEGEND
State transition of the HMM
Emission from the active state of the HMM
q_t Active state of the HMM at iteration t
O_t Observed outcome at iteration t

Figure 4.2: Trellis Graphical Representation of Observed Sequence from an HMM

4.2.2 Calculating Alpha Updating Parameters (Forward Algorithm/Procedure)

The process of computing alpha parameters in the Baum-Welch is known as a forward algorithm. It is sub-divided into two phases, namely a base phase and an induction phase. The base phase involves computing the alpha value of the first iteration (see Equation 4.1), while the induction phase computes the alpha values from the second iteration through to the last (see Equation 4.2). It is for this reason that the operation involving the computation of alpha values is referred to as a forward procedure.

$$\alpha_1(j) = \pi_j b_j(O_1) \tag{4.1}$$

$$\alpha_{t+1}(j) = b_j(O_{t+1}) \sum_{i=1}^{N} \alpha_t(i) a_{ij} \tag{4.2}$$

It is evident from Equation 4.2, that computing an alpha value for a given iteration in the induction phase makes use of the alpha value of the previous iteration, which is why the Baum-Welch algorithm is categorized as a dynamic programming algorithm. This same trend is applied in the computation of Beta parameters.

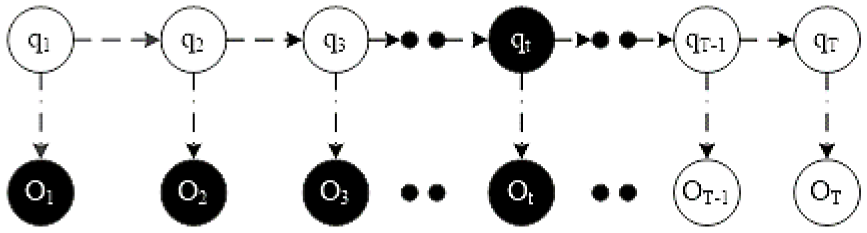


Figure 4.3: Interpretation of Alpha Parameter from a Trellis Diagram

In other words, alpha represents the probability of making observations up to the current iteration given that the active state of the HMM is set to a certain value (see Figure 4.3). The mathematical expression to illustrate this is presented in Equation 4.3.

$$\alpha_t(j) = P(O_1, O_2, \ldots\ldots, O_t; q_t = S_j \mid \lambda) \tag{4.3}$$

4.2.3 Calculating Beta Updating Parameters (Backward Algorithm)

Computation of beta parameters is often referred to as the backward procedure. The two procedures (forward and backward) are independent of each other. While it is often implemented second, there is no specific reason for implementing the backward procedure as a second step aside from the convenience that is associated with this order.

The backward procedure is similar to the forward procedure in that it also has a base phase and an induction phase. The base phase involves setting the beta value of the last iteration, while the induction involves computing beta values from the second last iteration to the first (hence the term backward procedure). Equation 4.4 is invoked in the base phase of the backward procedure, while Equation 4.5 is invoked within the induction phase.

$$\beta_T(j) = 1.0 \tag{4.4}$$

$$\beta_t(j) = \sum_{i=1}^{N} \beta_{t+1}(i) a_{ji} b_i(O_{t+1}) \tag{4.5}$$

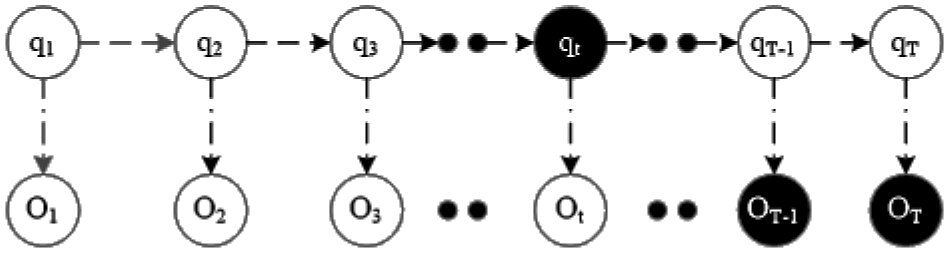


Figure 4.4: Interpretation of Beta Parameters from a Trellis Diagram

In essence, beta parameters represent the probability of having observations made after the current iteration that match actual observed outcomes, given that the activated state of the HMM is assumed to be one of the possible hidden states (see Figure 4.4). Equation 4.6 represents this description in mathematical terms.

$$\beta_t(j) = P(O_{t+1}, O_{t+2}, ..., O_T; q_t = S_j \mid \lambda) \tag{4.6}$$

4.2.4 Probability of Observations given HMM Parameters

Sometimes, a modeller is interested in knowing the probability of having the HMM in a given hidden state and being able to see the observations made. This value can be obtained for any iteration using computed alpha and beta values. Alpha represents the chance of making observations up to the current iteration, while beta represents the probability of having the observations after the index of the current iteration. Therefore, alpha represents one part of what is required, and beta represents the other—both the alpha and beta values are required for an iteration. This "AND" operator in probability theory translates into a product, generating Equation 4.7.

$$P(O, q_t = S_j \mid \lambda) = \alpha_t(j)\beta_t(j) \tag{4.7}$$

Equation 4.7 assumes that the HMM has an active state of j. Equation 4.7 transforms into Equation 4.8 if it is assumed that the HMM can take on any of the possible hidden states in iteration t. The possibility of the HMM having any of the possible hidden states as the active state introduces an "OR" probability operation, which, in turn, translates into a summation mathematical operator. This explains the introduction of the summation in Equation 4.8, which is an addition to Equation 4.7.

$$P_t(O \mid \lambda) = \sum_{i=1}^{N} \alpha_t(i)\beta_t(i) \tag{4.8}$$

At termination (i.e., the last iteration), the probability of obtaining the observations of the given the HMM parameters are computed using Equation 4.9.

$$P_T(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i)\beta_T(i) \tag{4.9}$$

It has been stated that the value of beta in the last iteration is always equal to one for all possible hidden states (see Equation 4.4), implying that Equation 4.10 holds true.

$$\beta_T(i) = 1.0 \tag{4.10}$$

Substituting Equation 4.10 into Equation 4.9 gives Equation 4.11. Equation 4.11 computes the probability of seeing all of the observed outcomes from the assumed HMM parameters. This value is computed at the end of the forward procedure in the Baum-Welch algorithm for use in subsequent steps.

$$P_T(O \mid \lambda) = P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{4.11}$$

Values obtained from Equation 4.1, 4.2, 4.4, 4.5, and 4.11 are utilized in the computation of zeta and gamma parameters.

4.2.5 Calculating Zeta Updating Parameters

The zeta parameter represents the probability of the HMM transitioning from state i to state j in iteration t. Equation 4.12 summarizes this description mathematically.

$$\xi_t(i,j) = P\left(q_t = S_i, q_{t+1} = S_j \mid O, \lambda\right) \tag{4.12}$$

As mentioned earlier, zeta parameters are derived from values obtained from the forward and backward procedures of the Baum-Welch algorithm. Equation 4.12 can be transformed into Equation 4.13 to summarize the math formula used for computing zeta values.

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{P(O \mid \lambda)} \tag{4.13}$$

Equation 4.13 represents the math formula for computing zeta values for updating the HMM.

4.2.6 Calculating Gamma Updating Parameters

Gamma values can either be derived from alpha and beta values or from computed zeta values. Equation 4.14 summarizes the mathematical equation for this.

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j) = \frac{\alpha_t(i)\beta_t(i)}{P(O \mid \lambda)} \tag{4.14}$$

Gamma values represent the probability of the HMM having its active state as state i when in iteration t. This is a useful metric in the updating of HMMs.

4.2.7 Updating HMM Parameters

The HMM parameters are updated using zeta and gamma values computed from the previous sections. The summation of gamma values for a specific iteration and state indicate the total number of transitions away from that state in that iteration. Note that this count includes transitions to the same state.

Equation 4.15 summarize the mathematical formulations used to perform these updates.

$$\pi_1^*(j) = \gamma_1(j) \tag{4.15}$$

Elements of matrix A are computed using the number of transitions from state i to state j divided by the total number of transitions from state i (including a transition that returns back to state i). Equation 4.16 summarizes this formulation concisely using values previously computed.

$$a_{ij}^* = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{4.16}$$

Elements of matrix B are computed by comparing the total transitions from state i when the observation related to the element being updated was the same as the observation made in that iteration to the total number of transitions from state i. Equation 4.17 summarizes this mathematically.

$$b_i^*(v_k) = \frac{\sum_{t=1}^{T} \gamma_t(i)\psi}{\sum_{t=1}^{T} \gamma_t(i)}$$

$$\textit{Where}: \quad \psi = \begin{cases} 1, \textit{ if } v_k = O_k; \\ 0, \textit{ Otherwise.} \end{cases} \tag{4.17}$$

4.2.8 Initializing of HMM Parameters

The convergence of the Baum-Welch algorithm is dependent on both the training data set used and the initial guess of the HMM parameters. It has been shown experimentally that uniform or random values for matrix A produce desirable results (Rabiner & Juang, 1993). Carefully selecting information related to the elements of the observation matrix B has also been found to improve the learning of HMMs (Rabiner & Juang, 1993). Implementing these two strategies improves the chance of the Baum-Welch algorithm converging at the global maxima.

4.2.9 Process Logic for Bayesian Updating

The entire process of updating HMM parameters has been described in the previous section as an iterative process that involves computation of parameters and using those for updating in an iterative fashion until the convergence criteria is met. This process is illustrated in Figure 4.5.

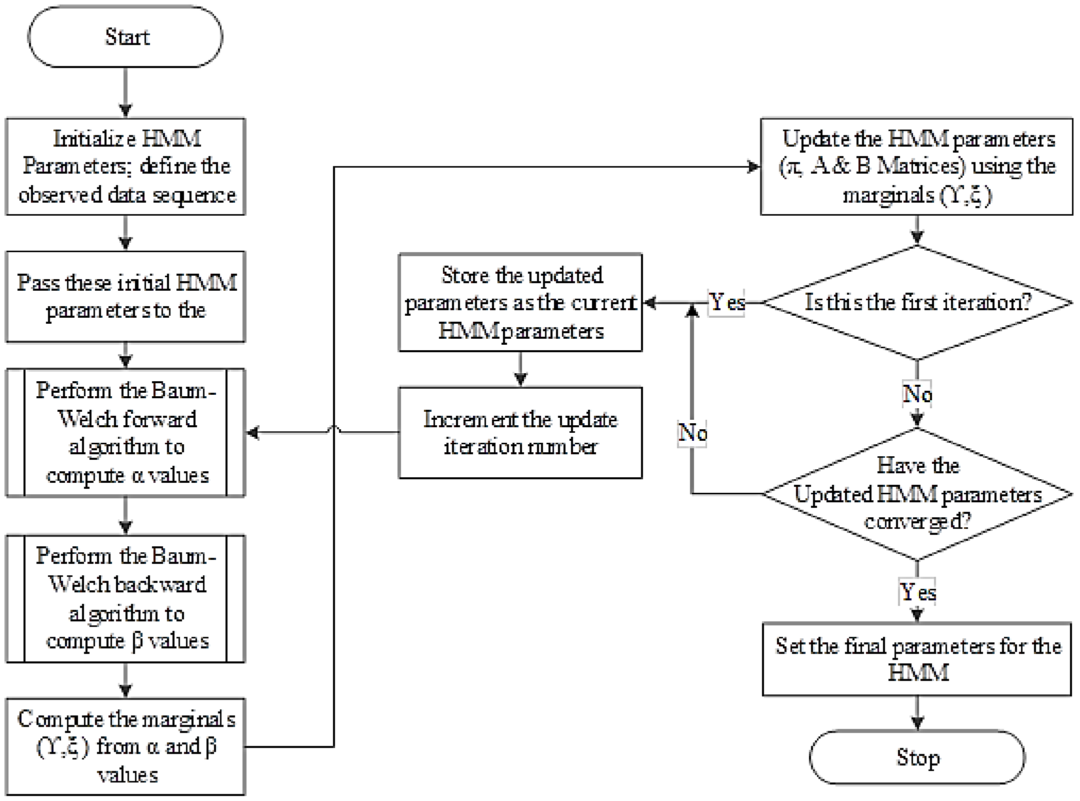

Start
Initialize HMM Parameters; define the observed data sequence
Pass these initial HMM parameters to the
Perform the Baum-Welch forward algorithm to compute α values
Perform the Baum-Welch backward algorithm to compute β values
Compute the marginals (ϒ,ξ) from α and β values
Update the HMM parameters (π, A & B Matrices) using the marginals (ϒ,ξ)
Is this the first iteration?
Yes
Store the updated parameters as the current HMM parameters
Increment the update iteration number
No
Have the Updated HMM parameters converged?
No
Yes
Set the final parameters for the HMM
Stop

Figure 4.5: Flow Chart for Markov Chain Updating Process

4.3 Development and Deployment of the Bayesian Updating HMM Prototype

4.3.1 Overview of the Whole Process

The implementation of the Baum-Welch algorithm was guided by the sequence summarized in the flow chart in Figure 4.6.

Literature was reviewed to identify the state-of-the-art related to this algorithm. A number of peer-reviewed journal papers and tutorials or lecture materials, especially within the computing science domain, were collected and examined. Common authentic elements from the information obtained from these sources were summarized into pseudo code. This was then translated into a computer program by implementing C# code in Microsoft Visual Studio development environment. This code was implemented as a dynamic link library project and later deployed within the Simphony.NET General Purpose Template (GPT) for testing.

Initial tests did not generate desired results because of errors that existed with the math formulas cited from literature. Further investigation and review of additional literature confirmed this. Correct formulae were identified and used to embellish the template. Tests subsequent to this demonstrated reasonable results. Details summarizing the testing in Simphony.NET and the results obtained are summarized in the following sections.

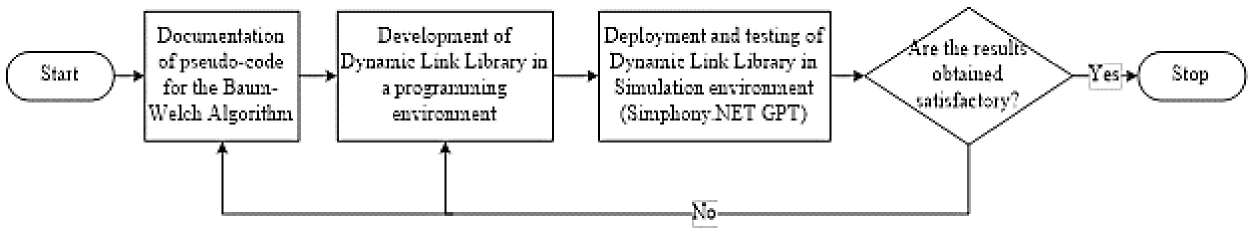

Start
Documentation of pseudo-code for the Baum-Welch Algorithm
Development of Dynamic Link Library in a programming environment
Deployment and testing of Dynamic Link Library in Simulation environment (Simphony.NET GPT)
Are the results obtained satisfactory?
Yes
Stop
No

Figure 4.6: Concept Layout of the Entire Development and Deployment Process for the Prototype

4.3.2 Design Specifications for the Prototype

The Baum-Welch algorithm was implemented as a dynamic link library to facilitate its implementation within other applications using the .NET framework. The core class is that which relates to the library, while the other classes (e.g., alpha, beta, gamma , and zeta) facilitate the storage of information and implementation of procedures within the library. The library, used to perform the Bayesian updating, was designed and implemented in such a way that it is not constrained in the number of hidden states and possible observable outcomes it can process.

4.4 Case Study: A Chicken-Egg Problem

Markov models are best suited for modelling systems that have a finite number of states and possible outcomes. Each state or outcome typically repeats after a variable number of iterations. To test implementations of Bayesian updating of hidden Markov models, a system that possesses these inherent processes and characteristics (i.e. generates sequential data) must be identified. The famous chicken-egg example is selected for this purpose because of its simplicity and relatability. Also, it is a low dimensionality problem (i.e., it involves 2D vectors and 2x2 matrices) that is suitable for illustrative purposes. The following example has been adapted from Frazzoli (2013).

4.4.1 Chicken-Egg HMM

In this problem, observations are made at the same time for a series of days to determine whether or not a specific chicken has laid an egg. This implies that there are only two possible observable outcomes (see Equation 4.18).

$$\begin{aligned}&\textit{Possible observable outcomes} = \{E, N\}\\ &\textit{Where}:\\ &E = \textit{Egg observed (chicken laid an egg)}\\ &N = \textit{No egg observed (chicken did not lay an egg)}\end{aligned} \tag{4.18}$$

Based on the identified possible outcomes, common practice is to leave the decision of the number of possible hidden states (i.e. states that the chicken transitions through that are not visible) to the discretion of the analyst. For convenience, it can be assumed that the chicken transitions between two states: state 1 (S1) and state 2 (S2). If a chicken is in either S1 or S2, there is a probability that is can either lay and egg or not. Figure 4.8 illustrates the hidden states for the chicken and their relation to the possible observed outcome.

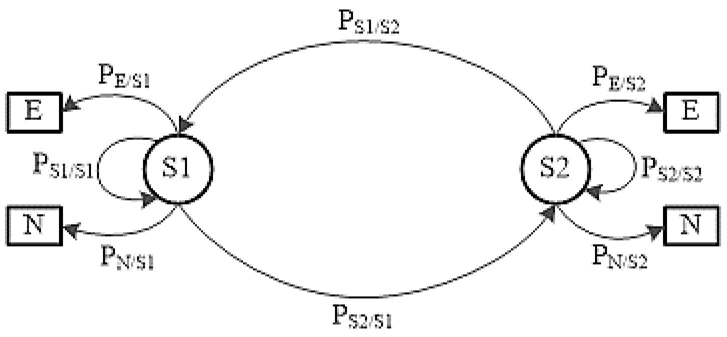


Figure 4.7: Graphical Representation of the HMM for the Chicken-Egg Problem

The notation used in the diagram of the Markov model can be interpreted as follows:

- PE/S1 and PN/S1 represent the likelihood of the chicken laying an egg and not laying an egg respectively, given that it is in hidden state S1.

- PE/S2 and PN/S2 represent the likelihood of the chicken laying an egg and not laying an egg respectively, given that it is in hidden state S2.

These arrows and notation are summarized in a concise mathematical form using an observation matrix.

- PS1/S1 and PS2/S1 indicate the chance of the chicken's hidden state returning to S1 and transitioning to S2, respectively, given that its current hidden state is S1.
- PS2/S2 and PS1/S2 indicate the chance of the chicken's hidden state returning to S2 and transitioning into S1, respectively, given that its current hidden state is S2.

These four notations indicated on the arrows are summarized concisely into a transition matrix.

For this problem, it is assumed that we observe the egg laying habits of the chicken for 2 weeks (14 calendar days) and make the following observations: {N, N, E, N, N, E, N, N, E, N, N, E, N, N}. The objective is to use the implemented developments of the Bayesian updating hidden Markov model within the Simphony General Purpose template to estimate the hidden Markov model that matches the observed data best.

The first step in estimating a Markov model using a Bayesian approach involves determining the inputs. In this case, initialized parameters (e.g., prior state density matrix, observation matrix, and the transition matrix) represent the inputs to the process. Rabiner & Juang (1993) suggested using uniformly distributed probabilities for initializing the prior state density matrix and the transition matrix. Since these values are unknown, and adopting this approach assumes that every possibility is equally likely to occur. However, Rabiner & Juang (1993) have also suggested making reasonable assumptions during the initialization of the observation matrix. In this particular case, and assumption is made that the chicken has a higher tendency to lay eggs when it is in S1 and a high tendency not to lay eggs when it is in S2. These assumptions result in parameters of the hidden Markov model that have initial values summarized in Equation 4.19.

$$\pi = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$

$$A = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix} \qquad A = \begin{bmatrix} \quad & \quad \end{bmatrix} \qquad B = \begin{bmatrix} \quad & \quad \end{bmatrix}$$

$$B = \begin{bmatrix} 0.3 & 0.7 \\ 0.7 & 0.3 \end{bmatrix} \tag{4.19}$$

$A = [a_{ij}]$ $B = [b_{ij}]$ $A^* = [a_{ij}^*]$ *updated $B^* = [b_{ij}^*]$

4.2 Simphony GPT Discrete Event Simulation Model

The next set involves creating a Simphony simulation model using the General Purpose Template that encapsulated the developed Bayesian Markov Updating algorithm. This was accomplished using a “create,” an “execute,” a “counter,” and a “destroy” modelling element. These elements were connected in sequence using directional arrows (see Figure 4.8).

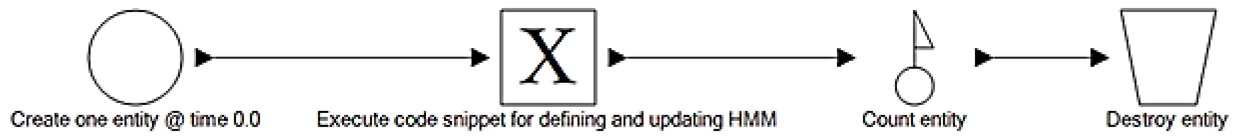

X
Create one entity @ time 0.0
Execute code snippet for defining and updating HMM
Count entity
Destroy entity

Figure 4.8: Simphony Model Layout for creating and updating HMM

The Bayesian Updating library was then encapsulated within the “execute” modelling element. C# code was then written within the formula editor of the “Expression” property for the “execute” element. Details of the code snippet embedded within the “execute” element is summarized in Appendix B.

After building the simulation model, it is reviewed for integrity. Then, the model is executed. No duration was set within the model, and the create element was setup to fire out one entity at time zero. This entity is created when simulation commences and flows into the “execute” modelling element. When this entity is transferred into this “execute” element, it triggers the execution of the code snippet within this element (see Appendix B).

This prompts Simphony to create instances of the initial parameters of the hidden Markov model using matrix classes in Simphony. A collection of the observed outcomes for the chicken's egg laying pattern is also created. These are all passed as arguments to a constructor for a hidden Markov model defined within the Bayesian updating library. Procedures defined within the library are then invoked, which trigger the Baum-Welch forward algorithm, backward algorithm, computation of marginal parameters, and updating of the hidden Markov parameters in each iteration. These procedures are invoked iteratively until the updated hidden Markov model converges. Once this occurs, the library stores the final results and returns control to the "execute" element at the point at which it had left off. At this point, Simphony retrieves and writes the final results from the library to its trace window. A screen shot of the final result displayed within Simphony's trace window is shown in Appendix A.

4.3 Discussion of the Results Obtained

The sequential data observed in two weeks' time contains information that can be used to assess the validity of the Markov hidden model Bayesian updating results. The sequence was setup such that, on the first day, the chicken was observed as having not laid an egg (N). Thereafter, it was observed to lay an egg (E) every two days at indices 2, 5, 8, and 11, assuming that the sequence indices are zero-based. In total, it was noted that there were four instances that the chicken laid an egg and ten instances that the chicken did not lay an egg. Information about the hidden states of the chicken can be inferred by intuition from the sequence because the dataset is small in size and has low dimensionality.

First, the chicken seems to have laid eggs on fewer days within the two weeks of observation. Examination of the initial observation matrix entries reveals that the chicken spent more time in S2 because the first column of the observation matrix, which relates to the likelihoods of observing the chicken not laying an egg, has a higher probability (i.e. 0.7) of the chicken not laying an egg when in S2. This should translate in the estimated or updated state prior density matrix having a higher probability value associated with S2 compared to S1 (see Equation 4.20).

$$\pi^* = \begin{Bmatrix} 0.212 \\ 0.788 \end{Bmatrix} \tag{4.20}$$

Likewise, we can assume that the initial estimates for the observation matrix are representative of the true behavior of the chicken by tracking the transitions between hidden states. Observing no egg laid likely maps to the hidden state, S2, while observing an egg likely maps to the hidden state, S1. The transitions in the observed data are summarized in Table 4.1.

Table 4.1: Statistics of Transitions Observed in the Data Sequence for the Chicken-Egg Problem

		Current Observation	
		N	**E**
Next Observation	**N**	5	4
	E	4	0
N = No egg was laid on that specific day E = An egg was laid on that specific day			

From the numbers summarized in the Table 4.1, it can be inferred that when the chicken is in hidden state S2, there is a slightly higher chance for it to transition back into S2 compared to transitioning into S1. Also, when the chicken is in hidden state S1, it will almost always transition to S2 rather than returning to the same state (S1). This translates in the updated transition matrix with an S2-S2 element having only a slightly higher value than the S2-S1 element. On the other hand, the S1-S2 element would be expected have a higher value compared to the S1-S1 element. This trend is evident in the updated transition matrix result shown in Equation 4.21.

$$A^* = \begin{bmatrix} 0.339 & 0.661 \\ 0.444 & 0.556 \end{bmatrix} \tag{4.21}$$

The results obtained from running the simulation model coincide with our expectations derived from a manual analysis of the data. This confirms the validity of the Bayesian updating method and its implementation as described in this chapter.

4.4 Conclusion

The first iteration for prototyping an HMM Bayesian updating algorithm was successfully developed and deployed within a discrete-event simulation environment (i.e. Simphony.NET). This version of the prototype generates reliable results as demonstrated with the chicken-egg example provided in the chapter. Using a data set that was small in size allowed a manual intuitive approach was used to demonstrate the validity of the result produced by the prototype. More extensive validation tests of the prototype should be performed using more complex, large-scale datasets within robust applications, such as MATLAB, in which this algorithm has been implemented.

After these limitations have been overcome and extensive testing has been performed on the prototype, it is recommended that the procedures encapsulated within the dynamic link library can be migrated into the math libraries of the Simphony simulation system. Thereafter, modelling constructs can be developed within the General Purpose Template, based on these procedures, to allow modellers to embed Markov models within their models. It is envisioned that modelling and analysis of construction processes or phenomena that generate sequential data may benefit the most from these Markov procedures and from the modelling constructs embedded within simulation environments such as Simphony. Examples of such processes or phenomena include geological condition prediction for sub-surface or underground construction work (e.g. tunneling and road construction) and weather generation. Successfully incorporating the full version of the Baum-welch algorithm implementation would be a first step in enabling simulation studies to make use of machine learning algorithms.

CHAPTER 5

A SPECIAL PURPOSE SIMULATION TEMPLATE IN SIMPHONY.NET TO FACILITATE MARKOV-BASED SIMULATION MODELING WITH A TUNNEL SIMULATION

5.1 Introduction

Markov chain models have been used for the analysis of systems, operations, and processes across a variety of different domains. Satisfactory results have been produced when Markov chains have been applied to the right type of problems. This can be attributed to their ability to explicitly represent different possible states of a system or process and provide stochastic variation between these states. Providing an implementation of Markov chain models within a simulation environment enriches computational analysis that can be done from both a simulation and Markov perspective. While such work has been done, it has been completed from a computational standpoint rather than a simulation modeling perspective. This chapter presents a special purpose simulation template developed within Simphony.NET that facilitates Markov chain modelling together with other types of simulation. The work extends the state-of-the-art in the use of Markov models together with simulation models by providing both an integrated template within the Simphony modelling environment with dedicated modelling elements and a user interface for defining and modifying Markov chains. The developed Markovian template was then implemented within a tunneling simulation project to predict the excavation production rate for certain ground conditions.

5.2 Simulating Markov Chain Behavior

Simulating a Markov chain involves following the transitions and emissions using the current state and a randomly generated number at every step that the Markov chain is executed. When doing this, one has to make use of the prior state probability vector and the transition probability matrix, just as is the case in predicting its behavior. However, the manner in which each of these Markov parameters is used differs. In predictive analysis, both the prior state probability vector and the transition probability matrix are used in the computations at every step. However, in the simulation of the chain behavior, the prior state probability vector is only utilized in the first step (i.e., chain initialization) to determine the starting state of the chain. It is subsequently removed, and the transition probability matrix used in subsequent steps. When a Markov chain is in a given state, Si, its next state is only dependent on its state. In other words, the random transition of the chain into the next state will depend on the probability values in the ith row of the transition probability matrix. Following this process fulfills the requirements of the Markov property (i.e., memoryless property). As such, it is extremely difficult to determine beforehand in which state the chain will be in during a particular step, hence the need for simulation. It should also be noted that since, at every step of the chain, the state probability vector will be a row vector that corresponds to the current state of the chain, state probability vectors at any step will always be one of the row probability vectors that make up the transition probability matrix. This is summarized in Figure 5.1.

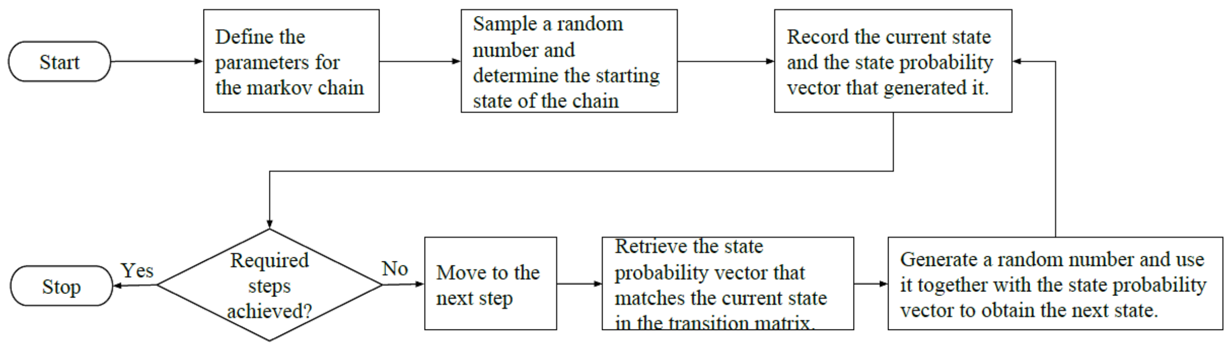

Start
Define the parameters for the markov chain
Sample a random number and determine the starting state of the chain
Record the current state and the state probability vector that generated it.
Required steps achieved?
Yes
Stop
No
Move to the next step
Retrieve the state probability vector that matches the current state in the transition matrix.
Generate a random number and use it together with the state probability vector to obtain the next state.

Figure 5.1: Logic for Simulating Markov Chains

5.2.1 Metrics for Characterizing Markov Chain Behavior

When predictions or simulations are performed on a Markov chain model, there is a significant amount of data that are generated, which can be used to make inferences about the process being studied using these chains and that can facilitate the characterization of chain behavior. A metric that is often used in literature in relation to Markov chain simulated behavior is the number of times each state was active.

5.2.1.1 State Activations

A measure that is important in Markov chains is the number of times that each state is active for a given simulation experiment. This count is based on the total number of steps used for simulating or predicting the Markov chain. It has been stated that the central limit theorem holds true for

Markov chain computations. Accordingly, the expected number of times that a state is likely to be active in a given experiment can be estimated as a product of the number of steps and the initial prior probability value for that state.

5.2.2 Summary on the Literature of Markov Chains

The subject of Markov processes and Markov chains has been studied and documented extensively in literature. Manual analytical computation and simulation of Markov processes and Markov chain models is a cumbersome process that is prone to error. Consequently, a number computer programs and computer-based tools have been developed to simplify computations of these processes or models. While their integration into traditional simulation environments would improve and facilitate modelling capabilities, many of these tools are not easily integrated into existing traditional simulation environments. The work presented in this chapter was an effort to address this challenge by implementing modelling elements and user interfaces that allow simple integration into the Simphony modelling environment. This template provides graphical modelling constructs that simulation modellers can use to define the parameters of a Markov chain in Simphony.NET and during simulation models at runtime.

5.3 Markov Chain Simulation Template and Elements

The majority of simulation environments that are available today provide graphical modelling constructs to facilitate modellers' efforts to visually represent the flow logic of the systems, operations, and processes that they have abstracted for purposes of simulation-based analysis.

As such, specific Markov chain elements were developed for use in the Simphony modelling environment general template allowing flexibility in the types of models that can be constructed. This provided the ability to define and emulate Markov chains in the Simphony.NET environment. The Markov elements were designed to have a one-to-one mapping between the constructs that appear in a graphical layout of a Markov chain model on paper and one developed using the envisioned Simphony.NET environment. This meant that Markov chain states would have to be represented by state modelling visualizations and, for the case of hidden Markov chains, would have to be represented by observations. Also, the transitions between states and emissions from states to observation symbols would have to be represented using directional arrows (i.e., relationships) with a probability value assigned to each. In addition, there was a need to have a higher-level modelling element, referred to here as the MarkovModel element, that would control and track Markov chain model parameters during model layout and simulation. Finally, there was a need to have a modelling element that would facilitate the modeller to trigger the Markov chain during the simulation. A modelling element, referred to as a MarkovTransition, was created for this purpose.

5.3.1 MarkovTransition Modelling Element

The MarkovTransition modelling element was implemented as a flow modelling element with one input and output port. One input property was provided to facilitate its association with a MarkovModel element. Each MarkovTransition element was setup to permit association with only one element. This ensures that every time a simulation entity arrives at its input port, it is transferred into the element causing it to trigger the Markov chain defined within the MarkovModel element to be stepped. Thereafter, the simulation entity is routed out of the element without delay. The visual appearance of the MarkovTransition modeling element is shown in Figure 5.2.

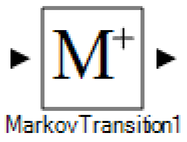


Figure 5.2: Graphical Appearance of the MarkovTransition Modelling Element in Simphony

Restrictions were embedded within this modelling element to prevent its insertion into a MarkovModel element. The MarkovTransition modelling element can be placed anywhere on the modelling surface at the scenario level and can be connected to any other general purpose modelling element in Simphony.NET.

5.3.2 MarkovModel Element

The MarkovModel element was implemented as a modeling element with no input or output ports (see Figure 5.3). It permits use only at the scenario-level together with a MarkovTransition modelling element and other general purpose template modelling elements. Nesting of

MarkovModel is not permitted, although multiple instances of this modelling element can be used within a simulation model to facilitate representation of multiple Markov chains in the process being simulated.

Figure 5.3: Graphical Appearance of the MarkovModel Element in Simphony

5.3.3 Stepping Markov Chain Using Code Snippets

When the MarkovModel element was being written, the method that implements all of the computations associated with stepping a chain were made public. In essence, they were exposed so that other general purpose template modelling elements, which are used within a model layout and in which code snippets can be embedded, could access the MarkovModel element and invoke this method. This provision was made for those that are comfortable and competent enough in writing code to advance simulation behavior.

5.3.4 Stepping Markov Chain Using Simulation Entities

Another option was provided for modellers that are not proficient in writing code to achieve certain desired behavior within their simulation models. This option involved writing a modelling element with one function: when a simulation entity is routed into it, the entity triggers the stepping of the Markov chain and is then routed out of the modelling element without delay. A property was included in this modelling element that allows a modeller to specify the MarkovModel element that it would be acting on. Besides inducing the Markov chain model to step, the "MarkovTransition" modelling element also keeps track of important statistics, such as the number of times that the Markov chain was stepped.

To summarize, the functions of MarkovModel element can be summarized as:

- Reading Markov chain parameters from the model layout encapsulated within it.
- Instantiating a Markov chain object at the beginning of each simulation run, which is made use of during the simulation run.

- The simulation modeller invokes the stepping of the Markov Chain through the MarkovTransition element.
- Invokes methods on the state and symbol modelling elements every step of the Markov chain, which triggers a collection of statistics that summarize their behavior during simulation.

5.3.5 Hidden States

The graphical construct used for the Hidden State modeling element is shown in Figure 5.4. An elliptical shape was used for this element. This element was setup to render the probability value of the element transitioning back to the same state on the graphical surface of the element. The state name is the same as the name of the modelling element and is displayed at the bottom of the element. This element also has an input port to which incoming transition relations are connected. The output port allow for connection of transitions and emissions leaving the state.

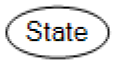


Figure 5.4: Graphical Appearance of the Hidden State (HS) Modelling Element

5.3.6 Observation Symbols

The Output modelling element was included in the MarkovModel to facilitate modelling of hidden Markov chains. Traditional Markov chains can also be created by developing a model layout that does not include observation symbols. A rectangular-shaped modelling element that has one input port and no output ports was used to represent the Output. The name of the symbol is displayed on the graphical display for the element rather than at the bottom of the element.

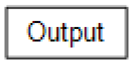


Figure 5.5: Graphical Appearance of the Output

5.4 Model Validation and Integrity Checks

The Simphony.NET simulation environment has a number of in-built validation or integrity checking services that a template developer can make use of to display guiding messages to the model developer. In the Markov chain special purpose template, checks enumerated provide for:

- *Hierarchical modelling and element location*: All modeling elements created were configured in such a way that they could only be used within the General Purpose Template. Another related restriction is that nesting of MarkovModel elements was not permitted.

- *Custom relationships:* Transitions and emissions were represented using a custom relationship. This relationship was written with one property—a probability value. A constraint was implemented at this property to ensure that only valid probability values can be entered by modellers (i.e., values from zero to one).

- *Markov chain model layout:* For Markov model layout to be considered valid, there should be at least two states and transitions from each of these states. An integrity check, which is initiated prior to the commencement of simulation, was used to ensure modeller compliance.

The described restrictions are especially useful for novices that are making use of the simulation environment and templates for their own modelling purposes.

5.5 Hypothetical Case Study

A hidden Markov chain model was used to demonstrate the functionality of the special purpose template that was created. A hypothetical chain comprised of two hidden states (HS1 and HS2) and two observable states (OS1 and OS2) was chosen. A graphical representation of the Markov chain is shown in Figure 5.6.

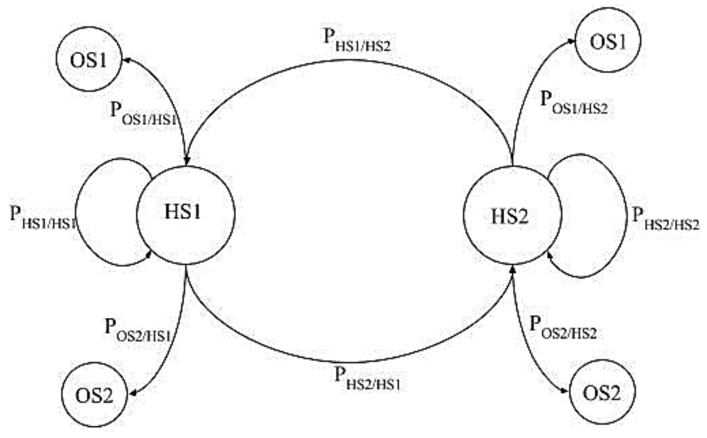


Figure 5.6: Graphical Layout of a Hidden Markov Chain Model

The equations in Figure 5.7 define the mathematical notation and values that were used for the prior state distribution and the transition and emission relation between the hidden states and observation symbols.

$$\Pr ior\ distribution\ (\pi) = \begin{Bmatrix} P^0_{HS1} \\ P^0_{HS2} \end{Bmatrix} = \begin{Bmatrix} 0.8 \\ 0.2 \end{Bmatrix}$$

$$Transition\ probability\ matrix\ (A) = \begin{bmatrix} P_{HS1/HS1} & P_{HS2/HS1} \\ P_{HS1/HS2} & P_{HS2/HS2} \end{bmatrix} = \begin{bmatrix} 0.3 & 0.7 \\ 0.9 & 0.1 \end{bmatrix}$$

$$Emission\ probability\ matrix\ (B) = \begin{bmatrix} P_{OS1/HS1} & P_{0S2/HS1} \\ P_{0S1/HS2} & P_{0S2/HS2} \end{bmatrix} = \begin{bmatrix} 0.5 & 0.5 \\ 0.2 & 0.8 \end{bmatrix}$$

Figure 5.7: Hidden Markov Chain Model Matrices

5.5.1 Simulation Setup

The Markov chain parameters used as inputs in the simulation model were based on the matrices and vector presented in the previous section. Parameters used to configure the simulation for experimentation are summarized in Table 5.1.

Table 5.1: Simulation Parameters Used in the Model

Parameter	**Value**
Seed Value	0
Run Count	100
Maximum Number of Steps @ Run	100
Stopping Criteria	100 simulation runs completed

5.5.2 Layouts of Simulation Model

The sketched graphical model and summarized parameter values of the Markov chain were transcribed into a simulation model in Simphony.NET using the modelling elements described. The simulation model layout was placed within the MarkovModel composite element labelled "MarkovModel1." Figure 5.8 graphically illustrates the Markov model layout of the MarkovModel element.

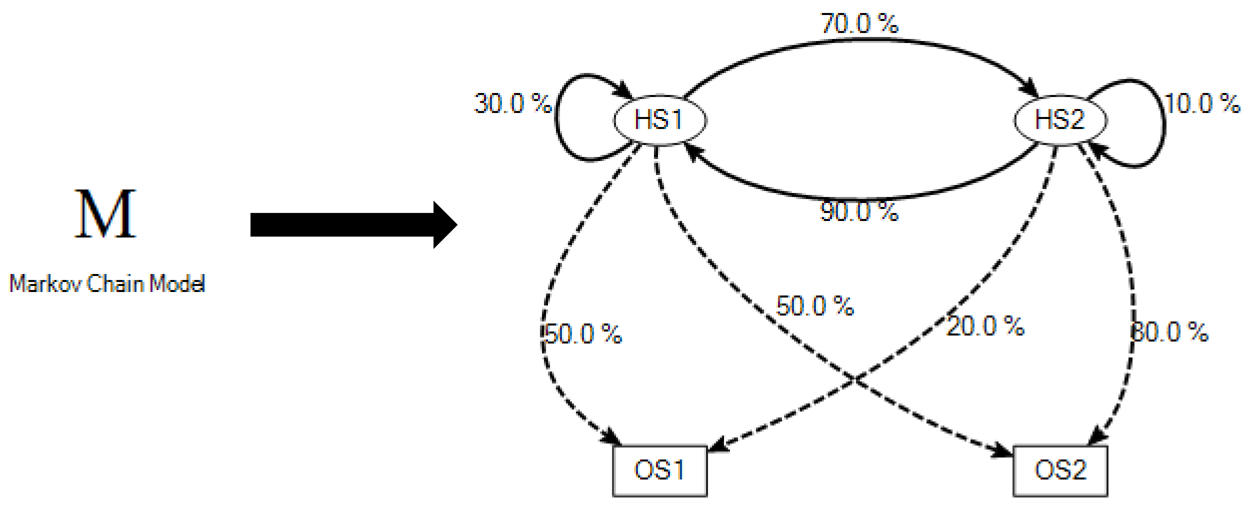


Figure 5.8: Markov Chain Model Layout Encapsulated in the Markov Chain Model Element

To make use of this Markov chain in a classical, discrete-event simulation model, the MarkovModel element, together with the MarkovTransition modeling element, were embedded in a general purpose template model (see Figure 5.9).

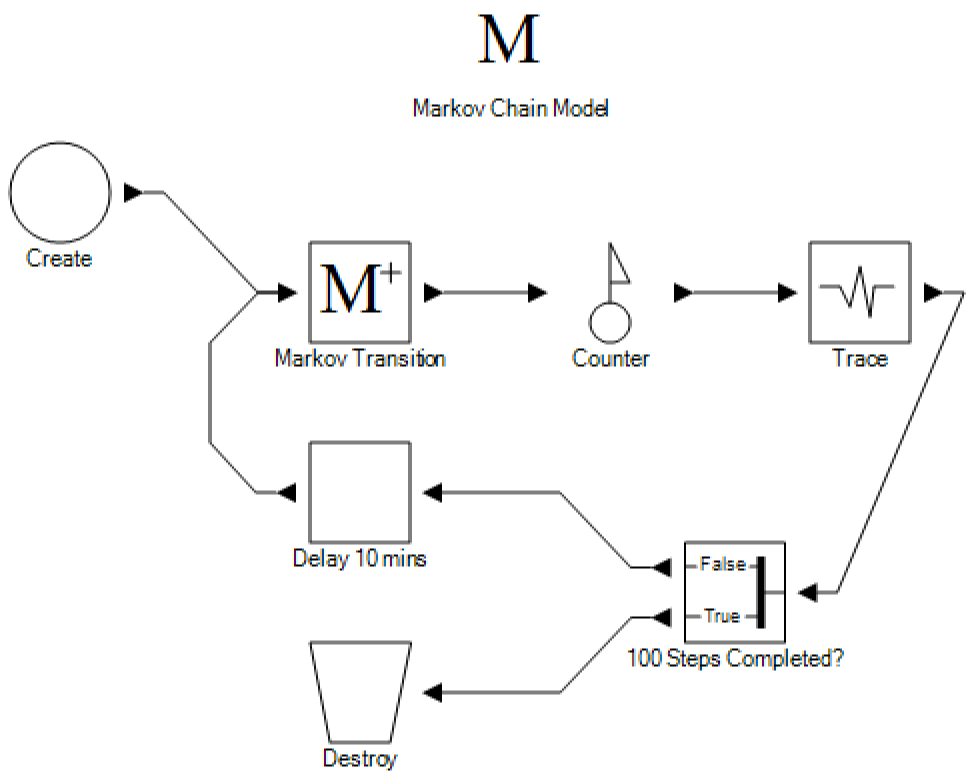


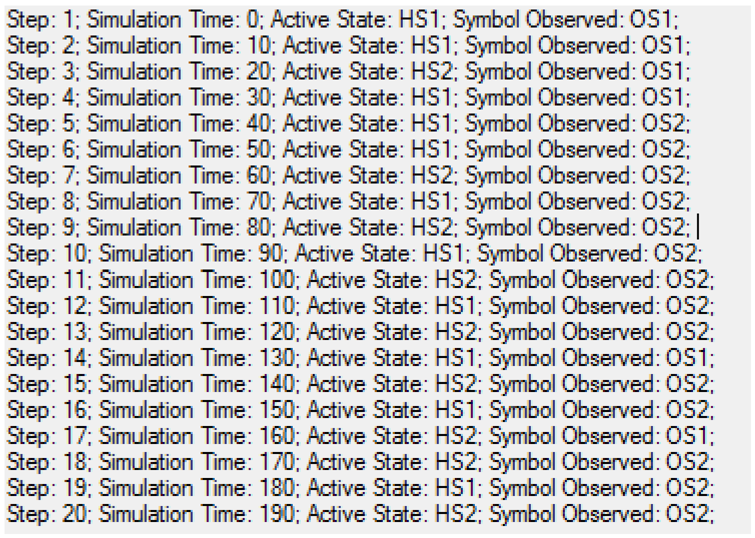
Figure 5.9: Markov Chain Elements together with General Purpose Template (GPT) Elements in Simphony.NET

In this model layout, the MarkovModel element contains the defined Markov model, while the MarkovTransition element steps (i.e., executes) the Markov chain every time that an entity flows through it. The model layout was configured in such a way that it causes the Markov chain to step every 10 time units for 100 steps before the simulation terminates. To achieve this behavior, one simulation entity is created at time zero by the element labelled "Create." The simulation entity is routed into the element labelled "Markov Transition," causing it to capture the "Markov Chain Model" element and step the Markov chain model encapsulated within it. The entity then flows through a Counter element and then into a Trace element, which captures the step number, simulation time, the active state, and the observed output. Thereafter, the simulation entity is routed into the Branch element, labelled "100 Steps Completed?" This checks to see if all 100 steps have been completed. If so, the element routes the simulation entity through the "True" branch and then into a "Destroy" element, where it is destroyed and terminating the simulation. Otherwise, if all 100 steps have not been performed, the simulation entity is routed through the "False" branch of the Branch element into a Task element labeled "Delay entity (10 units)." The simulation entity is delayed before it is routed back to the MarkovTransition element labelled "Markov Transition." The cyclic loop that involves the entity traversing the "Markov Transition," "100 Steps Completed?," "Counter," and "Delay entity (10 units)" proceeds until the Markov chain has been stepped 100 times.

The code snippet embedded within the Branch modelling element to perform the logical check that determines whether or not the stepping of the chain has been completed is summarized below:

```
return Count("Counter") == 100;
```

During the execution of the simulation model, certain information is generated, with certain information being tracked at the State, Output, and MarkovModel element level. The first type of result generated by the special purpose template is a trace log. The MarkovModel traces the parameters of the chain encapsulated within it together with logical time, active state, and symbol observed at each step (see Figure 5.10).



```
Step: 1; Simulation Time: 0; Active State: HS1; Symbol Observed: OS1;
Step: 2; Simulation Time: 10; Active State: HS1; Symbol Observed: OS1;
Step: 3; Simulation Time: 20; Active State: HS2; Symbol Observed: OS1;
Step: 4; Simulation Time: 30; Active State: HS1; Symbol Observed: OS1;
Step: 5; Simulation Time: 40; Active State: HS1; Symbol Observed: OS2;
Step: 6; Simulation Time: 50; Active State: HS1; Symbol Observed: OS2;
Step: 7; Simulation Time: 60; Active State: HS2; Symbol Observed: OS2;
Step: 8; Simulation Time: 70; Active State: HS1; Symbol Observed: OS2;
Step: 9; Simulation Time: 80; Active State: HS2; Symbol Observed: OS2;
Step: 10; Simulation Time: 90; Active State: HS1; Symbol Observed: OS2;
Step: 11; Simulation Time: 100; Active State: HS2; Symbol Observed: OS2;
Step: 12; Simulation Time: 110; Active State: HS1; Symbol Observed: OS2;
Step: 13; Simulation Time: 120; Active State: HS2; Symbol Observed: OS2;
Step: 14; Simulation Time: 130; Active State: HS1; Symbol Observed: OS1;
Step: 15; Simulation Time: 140; Active State: HS2; Symbol Observed: OS2;
Step: 16; Simulation Time: 150; Active State: HS1; Symbol Observed: OS2;
Step: 17; Simulation Time: 160; Active State: HS2; Symbol Observed: OS1;
Step: 18; Simulation Time: 170; Active State: HS2; Symbol Observed: OS2;
Step: 19; Simulation Time: 180; Active State: HS1; Symbol Observed: OS2;
Step: 20; Simulation Time: 190; Active State: HS2; Symbol Observed: OS2;
```

Figure 5.10: A Sample Trace Log in Simphony.NET Generated by the Trace Element

Each MarkovModel element creates a unique trace log, which is categorized based on its name to avoid mixing trace output in situations where there are multiple MarkovModel elements used in the same simulation model.

5.5.3 Simulation Results (Probability Stave Vector-PSV)

One of the results generated by the GPT simulating the hypothetical Markov chain model is the number of times that each state was active. Stacked charts were created for purposes of graphing this result. The result shows variation between simulation runs and always totals the total number of steps that the GPT was setup to execute. Altogether, these results demonstrate that this aspect of the Markov chain has been implemented accurately.

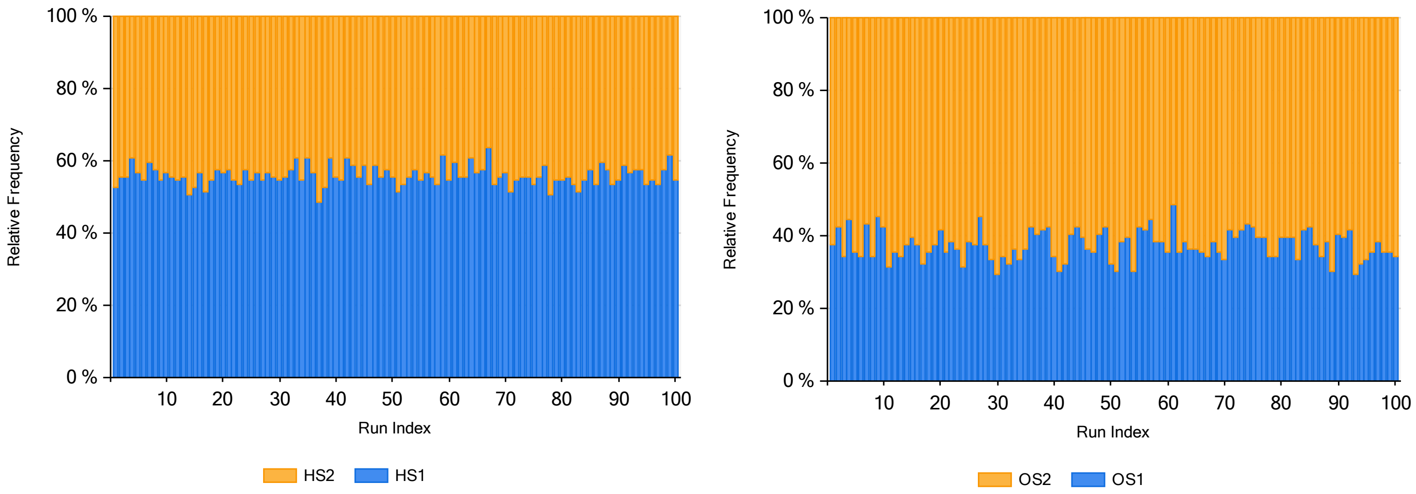


Figure 5.11: Number of State and Observation Activations Simulated for the Hypothetical Markov Chain

Simulation computations of a Markov chain are meant to strictly adhere to the memoryless property that is exhibited by all Markov processes. For this to hold true, the probability state vector of the Markov chain at each simulation step will always be one of the row vectors of the transition probability matrix. For the hypothetical example presented, the possibilities include: {0.3, 0.7}, and {0.9, 0.1}. From a visual inspection of the charts in Figure 16.0, PSV values plotted appear to coincide with the elements in the set of possible vectors.

5.6 Applications of Markov Chain Template in Simphony: Tunneling Project with Markov Prediction of Ground Conditions

5.6.1 Model Background and Details

The Markov chain special purpose simulation template was written to support the definition, prediction, and simulation of various types of Markov chains. As such, it can be applied as tool for automating the modelling and analysis of processes, operations, or systems for which Markov chain models are a best fit. Typical examples of such processes in construction include modelling geological conditions in tunneling, weather generation, modelling equipment breakdowns, and earned value analysis.

This case study displays the functionality of the developed Markov chain template and elements applied to a construction process. In tunneling projects, a key parameter that affects productivity is the excavation rate of tunneling. This excavation rate is depends on the ground condition encountered. The model below presents a hypothetical model that is based on the train cycle. First, it travels out to the Tunnel Boring Machine (TBM), where the liners are unloaded and the muck cars are loaded with spoils. Then, the train returns to the undercut, where the spoil is unloaded by the crane, and a new set of liners is loaded. Finally, the train waits to begin its next trip to the TBM.

The Markov elements in this model function to predict the ground conditions of the tunnel, which act as the outputs of the hidden Markov model. Each ground condition has an associated productivity that informs the amount of time it takes for each meter of excavation to occur. The model schematic is provided below:

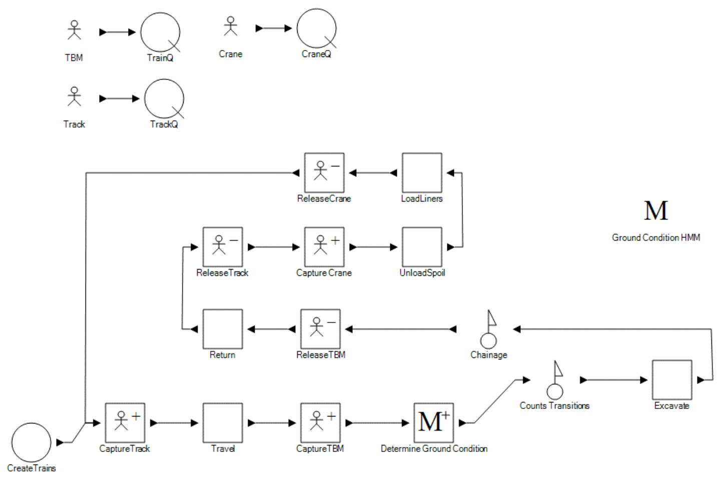


Figure 5.12: Tunneling Project Model Schematic with Markov Model Applied

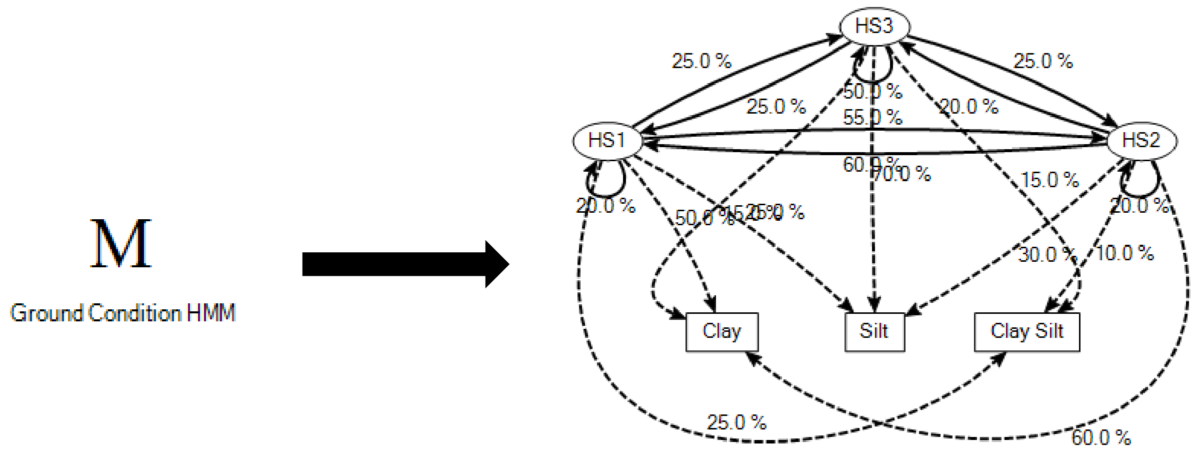


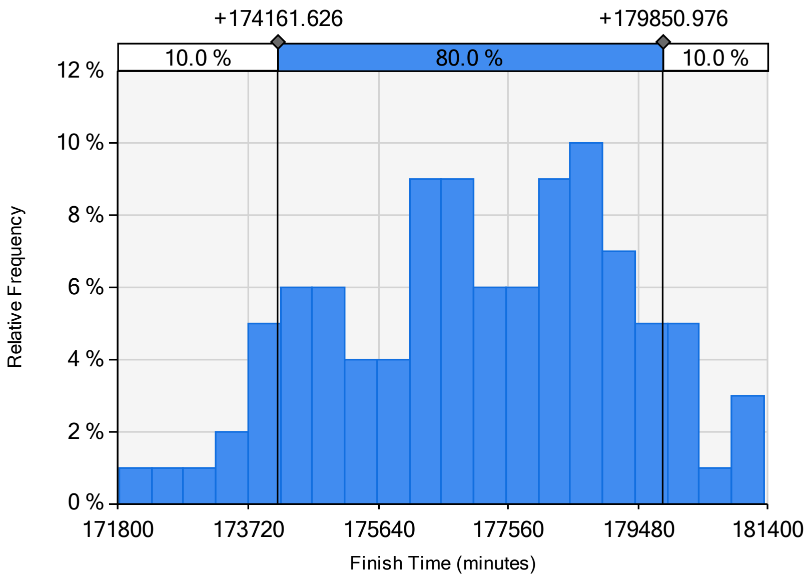
Figure 5.13: Markov Model Chain Visual Representation

As the entity approaches the excavation task, the geotechnical condition is determined based on the Markov chain in Figure 5.13. Depending on the ground condition expected, excavation productivity rate is sampled from the relevant distribution as per the details of the code below. The productivity rates provided are in minutes required to excavate 1 meter of material. Based on this, clay has a higher productivity rate compared to silt.

```
If GetMarkovOutput("Ground Condition HMM")="Clay"
    Return SampleBeta(2,2,45,130)
Else If GetMarkovOutput("Ground Condition HMM")="Clay Silt"
    Return SampleBeta(2,2,80,200)
Else If GetMarkovOutput("Ground Condition HMM")="Silt"
    Return SampleBeta(2,2,120,300)
```

The overall productivity in this case study model is highly dependent on ground conditions during excavation. It is important to note that the productivity rates provided are only used to illustrate the application of the Markov model to a tunneling simulation. They are not intended to be representative of actual data. To collect sufficient information for the tunneling case, 100 simulation runs were perfomed, each with a limit of 1000 meters of excavation.

5.6.2 Simulation Results



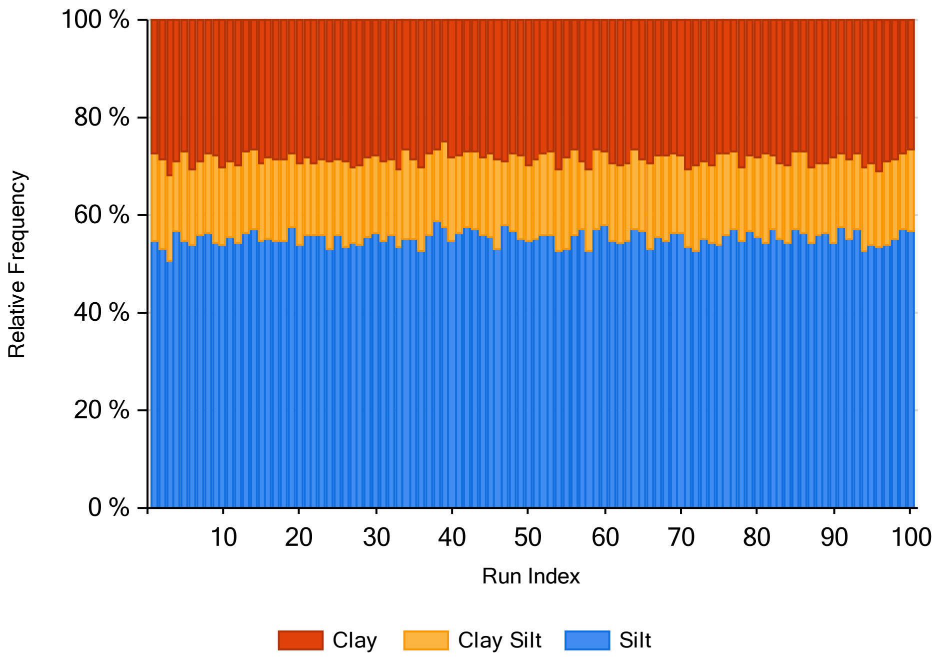


Figure 5.14: Results of Modelling Displaying the Histogram of Project Durations (top) and a Summary of the Observations of the Ground Conditions Activations.

5.7 Conclusions and Recommendations

A Markovian modelling system was implemented in the Simphony modelling environment and was successfully tested. The implementation has the capabilities to facilitate the definition of a Markov chain, perform simulations and predictions from the defined Markov chain, and track step-based data generated from computations performed on the defined Markov chain. The elements support modelling of both hidden Markov models and non-hidden Markov models (i.e., traditional Markov chains). The objective and benefits of automating Markov chain definition, computations, and related generated data tracking were achieved through this approach.

Five custom graphical modelling constructs (i.e., elements) were written within the Simphony.NET simulation environment to form a Markov template. The details and functions of these elements were discussed with assumptions and limitations noted. Simphony.NET's environment also provides features that were used by the template for visualizing the data generated from Markov computations. The modelling elements were tested using a simple model to illustrate how the chain steps throughout the simulation from Trace element outputs. Finally, the Markov template was integrated into a sample tunneling project model to demonstrate the ease of use and viability in predicting ground conditions in tunneling simulation.

CHAPTER 6

IMPLEMENTATION OF HMM AND BAYESIAN UPDATING IN A TUNNELING CASE STUDY

6.1 Introduction

Typically, the inputs for a simulation model are obtained from industry experts' best estimates in the form of deterministic values or, at best, a statistical distribution. For example, the productivity of excavation in tunneling simulation is often represented by a distribution, as there is a high-level of uncertainty for the duration required for each meter of tunneling. Inputs can be enhanced by obtaining actual data from previous, similar construction projects or as construction commences and using this information to improve the base data for the project. However, incorporating actual data into an existing tunneling model requires additional techniques. Ultimately, it would be beneficial to have the ability to take new information and data and use it to improve already developed models.

The basis of Bayesian updating is to improve the legitimacy and accuracy of predictions through the incorporation of newly acquired information. Bayesian updating is reliant on data collected in the real system being studied. These data are input into algorithms that can then improve the prediction models being used. In HMMs, the set of observations are a sequence of various outcomes, and traditional Bayesian updating techniques are not particularly amendable to this type of observation. With HMMs, both the unique instances in the data sequence and the transition from one outcome to the next must be accounted for while updating the model or chain. The observations and transition information become required inputs in the updating process. It has been shown that, although analytic methods cannot be applied for updating and training of HMMs, many numerical techniques can be applied to develop reasonable updates to HMMs, including the Baum-Welch Algorithm (Rabiner, 1989). This chapter demonstrates the use of the Baum-Welch algorithm for the purposes of training HMMs. The Baum-Welch method was selected for implementation in this study due to ease of application and to its ability to be implemented into the Simphony modelling environment.

6.2 Objective

The objective of this chapter is to use a project as the basis for planning the next phase of a project. Data from project A (107th/111th) acts as the historical, similar project with which the planning model for Project B is trained. Real construction data from Project B acts to update the simulation model for 100m increments of excavation. At each point the model is updated with construction data from Project A as well as real construction data from Project B to predict the remaining excavation. These updated models are then compared to the results of the original model, which is comprised of data from only Project A as well as the actual construction durations for 500m of Project B.

6.3 Case Study

6.3.1 Background

The case study project used in this study was the Downtown Intensification project, which is a three-phase, drainage improvement project in downtown Edmonton, Alberta, with work conducted by the City of Edmonton Drainage Group. This study focuses on two of these phases, consisting of two tunnels: a 1km-long tunnel along 107th Street from 111th to 105th Avenue (Project A) and a 500m-long tunnel along 105th Street from 105th to 100th Avenue (Project B). Each tunnel was a 2340mm diameter storm tunnel, to be completed using Tunnel Boring Machine (TBM) excavation. Both tunnels were constructed using a shielded TBM (M100) with a one-pass, pre-cast concrete segmental liner system for both temporary and permanent support. The M100 cuts a nominal tunnel diameter of 2540 mm, and pre-cast segmental liners are placed as the TBM advances. The construction method involves advancing the tunnel by jacking against the installed concrete liner and erecting and expanding/jacking a ring of four, pre-cast segments tightly against the exposed soil approximately every 1m of advance along the excavated tunnel. Based on the boreholes data that was completely prior to construction, the majority of tunneling is expected to be in clay till (with inter-till sand zones), a mix of sandy clay, and the possibility of pockets of sand.

6.3.2 Data Preparation and Model Inputs

Beginning in September 2015, TBM excavation took place for 978m of tunnel along 107th street, completing this distance at the end of August 2016. Through this phase, geotechnical conditions were monitored on a daily basis along with daily excavation progress and shift length. The project data including details of the geotechnical conditions encountered are summarized in Table 6.1.

Table 6.1: Project Data from Project A (978m)

Ground Conditions	Count (m)	Length of Persistance (m)			Effective Duration (hr/m)	
		Average	Min	Max	Average	Standard Deviation
Clay	207	13.8	3	44	2.16	0.16
Clay Sand	712	37.5	3	174	2.10	1.45
Sand	59	6.6	2	10	2.34	2.07

The effective duration statistics were calculated based on the shift time divided by the meters excavated each day. For an 8-hour shift where 4m were excavated, the effective duration per meter of excavation would be 2 hours. Calculating the duration for the excavation of one meter was required for modelling the tunneling construction. The most commonly encouraged ground condition was clay sand with 712m of excavation length categorized as such. It should be noted that, while all ground conditions have very similar average durations, the level of variance of each conditions was quite variable.

Data for Project B are summarized in Table 6.2. No sand was observed for Project B through the excavation of 521m. The dominant ground condition for this phase of the project was clay with a section of clay sand for 66m. This data set displays the difference in ground conditions that can be found for very similar projects that are in close proximity to the previous project.

Table 6.2: Project data from Project B (521m)

Ground Conditions	Count (m)	Length of Persistance (m)			Effective Duration (hr/m)	
		Average	Min	Max	Average	Standard Deviation
Clay	455	226	187	265	2.41	1.07
Clay Sand	66	66	66	66	2.35	0.59
Sand	0	0	0	0	0.00	0.00

Table 6.3 **summarizes actual project duration data for Project B at 100m intervals**. This **will serve as the baseline** for **comparing the effectiveness of simulation runs.**

Table 6.3: Summary of Project B Actual Data

Chainage	0-100m		100m-200m		200m-300m		300m-400m		400m-500m		Total (500m)	
Statistic	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation
Clay	2.10	0.43	2.18	0.59	2.73	1.61	2.33	0.69	2.70	1.51	2.41	1.07
Clay sand	N/A	N/A	2.03	0.06	2.44	0.65	N/A	N/A	N/A	N/A	2.35	0.58
Actual Duration (hours)	211		208.50		232.50		229.5		270		1152 hours or 144 days	
Total Actual Duration (hours)	211		419.5		652		881.5		1152			

6.3.3 Model and Assumptions

A tunneling construction Simphony model was developed based on a proposed model that was detailed in "Construction Simulation-An Introduction Using SIMPHONY" (AbouRizk, Ekyalimpa, Hague, 2016). The baseline model used in the textbook was originally created to model a sanitary sewer being constructed in Edmonton. Due to the similar nature of the case study in this chapter, the proposed model was used as a baseline to provide a validated model with which to conduct tests. This model was then embellished with the Markov Chain elements and updated project parameters for the case study. The functionality of all modelling elements used from the Simphony modelling environment are provided in Appendix 6.C. The model schematic used for the case study is provided in Figure 6.1.

The model begins with two trains being created by the Create element. Both train entities move into the Capture Track element, where the resource is captured for one train entity while the other waits in the Track file element. The train entity, which has captured the track, moves to the Travel task where the travel of the train is modelled to the tunnel face taking a duration of 5 minutes. The TBM resource is then captured, and the entity moves into the Markov transition element where the ground condition is predicted. Once the Markov model has transitioned and produces an output, which does not have a duration, the Counter counts the transition of the chain. The entity then moves into the Excavate element, which models the excavation of 1m of tunnel construction. The excavation duration is dependent on the ground condition (i.e., output) of the HMM chain. The pseudo code sample below provides the details of how the excavation duration is determined:

```
If Ground Condition HMM has an output of "clay"
    Duration = Sample of Normal Distribution for clay (Average duration for clay, standard deviation for clay)
If Ground Condition HMM has an output of "clay sand"
    Duration = Sample of Normal Distribution for clay sand (Average duration for clay sand, standard deviation for clay sand)
If Ground Condition HMM has an output of "sand"
    Duration = Sample of Normal Distribution for sand (Average duration for sand, standard deviation for sand)
```

After excavation, the Chainage Counter element counts that 1m has been excavated. The TBM is then then released, and the train then returns to the working shaft where the track is released (the second train entity currently waiting in the Track file then claims the track. The crane is captured, and spoils are unloaded and liners are loaded, which have durations of 6 and 15 minutes, respectively. At this point, the entity flows back to the beginning, and the process begins again.

The Task elements shown on the right side of the model schematic are used to store the distributions for the durations of each ground condition. These tasks are where the Bayesian Updating occurs through the distribution fitting interface provided within Simphony.

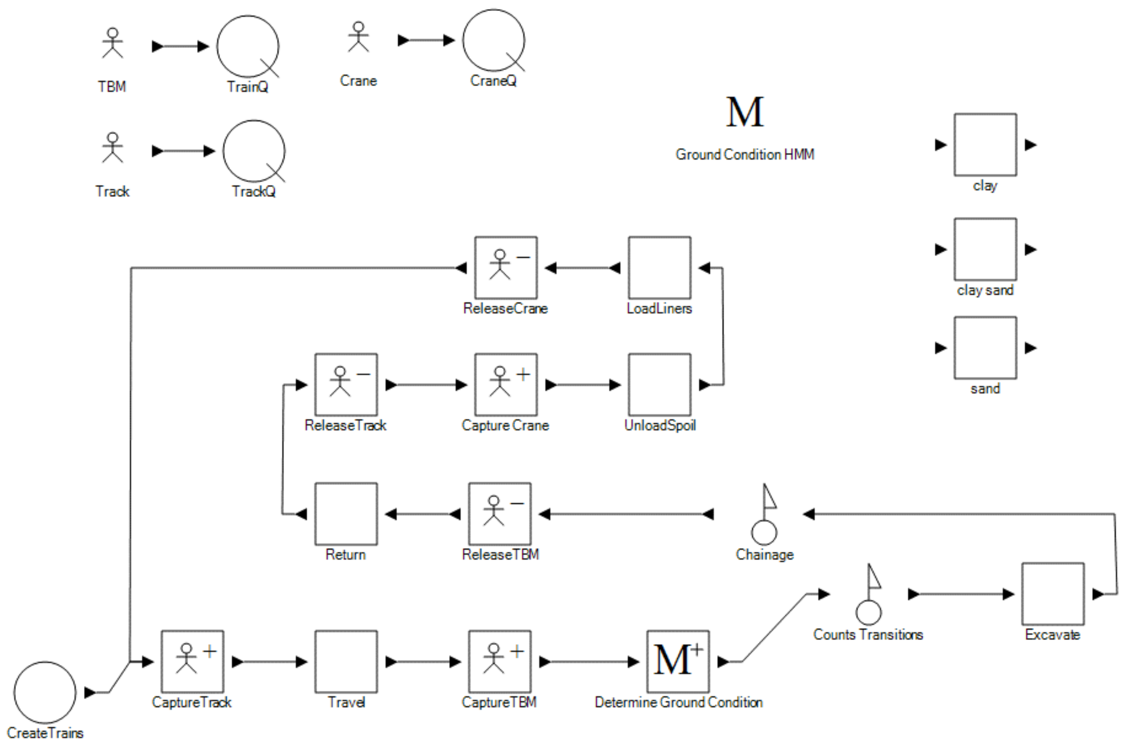
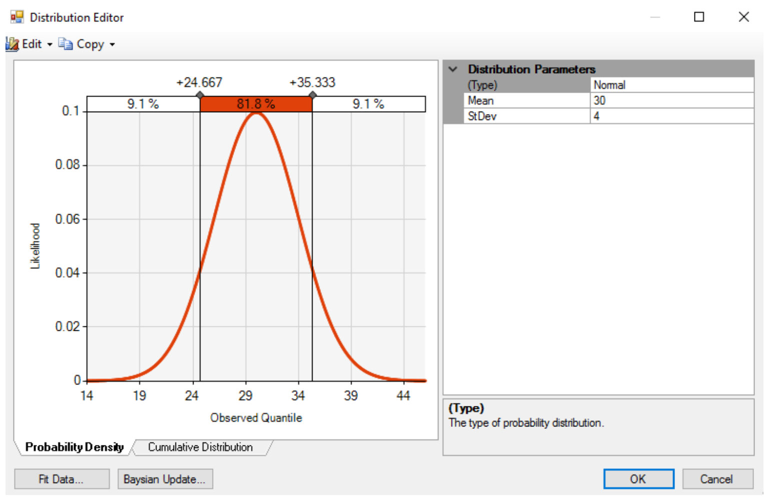


Figure 6.1: Case Study Tunneling Model Schematic

The Baum-Welch algorithm implementation into Simphony was developed in the previous chapter. The Bayesian updating functionality integration into Simphony was a feature that was already within the software. However, the user interface in Figure 6.2 was added into the distribution fitting interface for ease of use .



Distribution Parameters	
(Type)	Normal
Mean	30
StDev	4

(Type)
The type of probability distribution.

Figure 6.2: User interface for Bayesian Updating

The above window allows the viewing of the current distribution that is selected for a task's duration and, most importantly, allows the Bayesian updating of the distribution using imported data. The data is selected under the "Bayesian Update" tab and must be stored in a comma separated values (csv) file. The resulting distribution is then provided in the same window with updated parameters shown on the right. All Bayesian updates for the case study runs were done using this functionality.

Certain assumptions were made in modelling the tunneling construction for the case study project to allow for clear implementation of the updating and prediction techniques. Applying Bayesian updating to other distributions, such as beta distributions or triangular distributions, is much more complex and requires advanced analytical techniques that are outside of the scope of the contributions of this thesis. For this reason, all duration data used was assumed to be normally distributed, as the Simphony modelling environment has existing capabilities to handle this type of distribution. As previously discussed, additional user interface functionality was added to allow for straightforward Bayesian updating with effective visualization.

Other assumptions made to improve the transparency of the effects of the changing ground conditions in the simulation model were to assume constant train travel times to and from the tunnel face. This assumption is reasonable as the duration used would be overestimated at the beginning of the tunnel and underestimated for the tail end of the tunnel. Overall, the train travel time would be unlikely to dictate major changes to the project duration. Keeping the train travelling constant allowed for a more detailed focus on the effects of ground conditions on duration. The final assumption made during modelling is that the installation of liners is inherent in the excavate element, occurs in parallel with the described tunneling process, and, therefore, does not delay excavation.

A total of 6 simulation runs were completed to assess the effectiveness of the implementation of HMM techniques and Bayesian updating in the tunneling model. The tunnel model above was trained using the Project A data provided in Table 6.1 to develop the baseline HMM using the implemented Baum-Welch algorithm as illustrated in Figure 6.3.

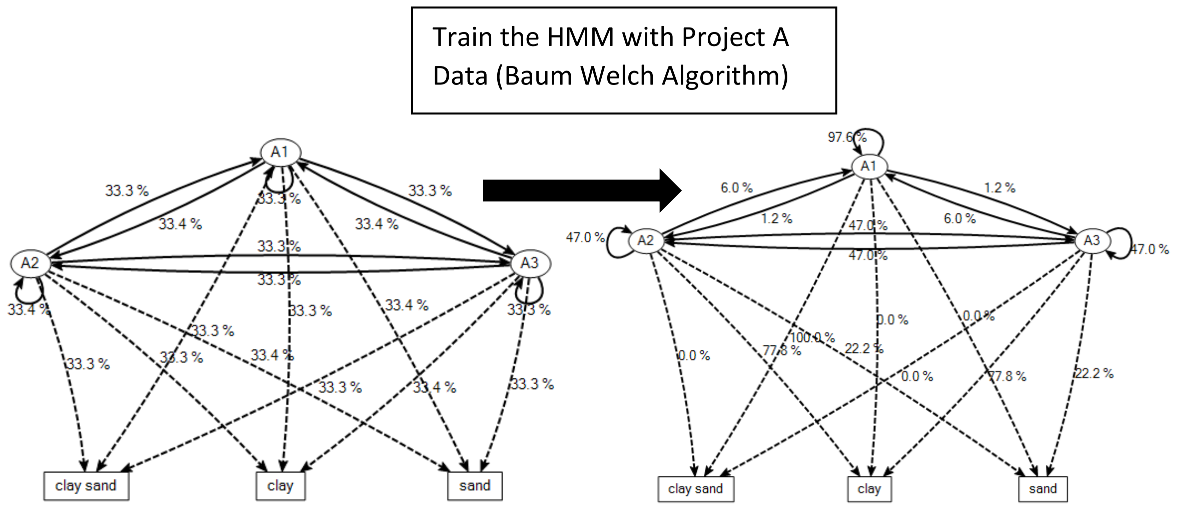


Figure 6.3: Baum-Welch training step of HMM

The code snippet below provides a summary of the ground condition excavation rates used from Project A data as per Table 6.1.

```
If GetMarkovOutput("Ground Condition HMM")="clay"
                    X = SampleNormal(2.156585613, 0.162328535)
          Else If GetMarkovOutput("Ground Condition HMM")="clay sand"
                    X = SampleNormal(2.097527995,1.447476952)
          Else If GetMarkovOutput("Ground Condition HMM")="sand"
                    X = SampleNormal(2.339795918,2.070096921)
          Else
                    Throw New NotSupportedException("Unknown Ground Condition!")
          End If
```

6.3.4 Simulation Runs and Processes

A simulation run was completed to represent actual project updates at 100m intervals along the 500m project length. The basis of the modelling effort was to mirror how the actual construction project planning and control would occur. Prior to project commencement (Project B), a model is created with basic assumptions and best guesses as to what types of productions and durations may occur. This model can be further improved using similar past projects or expert knowledge. For this case study, Project A data were used to "train" the Project B model as explained above. At this stage, the first run (Run #1) was completed to simulate the tunneling of 500m to predict the duration of Project B. At this point, actual construction would begin, and real data would be collected. After 100m of tunnel was completed, further modelling would be completed to predict the duration for the remaining 400m of tunnel. Data obtained during the first 100m of actual tunneling, including ground conditions encountered, excavation rates, and shift hours, would be used to update the model using the techniques described here. Once the HMM chain and respective production rates for each ground condition are updated, as previously described, the model is ready to predict the remaining 400m of the project in the second run (Run #2). This process is repeated for each 100m interval using all previous Project A data together with the up-to-date actual construction data for Project B until all tunneling construction is complete. For every 100m of actual construction completed, the model should become more accurate. Indeed, the predictions should, ideally, converge to actual construction duration. Each run is analyzed by taking the actual duration to date and adding the mean predicted duration for the remaining tunnel length as shown below. Each run number was performed 1000 times to develop a meaningful sample of results.

Table 6.4: Simulation model run details

Run Number	Simulated Length	Actual Data Length
Run #1	500m	0m
Run #2	400m	100m
Run #3	300m	200m
Run #4	200m	300m
Run #5	100m	400m
Run #6	500m	0

As presented above, the final run (Run #6) provides a good indication of how effective the model and updating process is by simulating the entire project using all 500m of actual construction data. This can be thought of in the case of predicting another 500m of tunnel or simply a comparison of the model to the actual construction duration achieved.

6.3.5 Results

The results of all six runs completed are summarized in this section. Simulation outputs are provided to display the HMM chain, ground conditions predicted, and distribution of total run durations for each run. Figure 6.4 illustrates the evolution of the HMM from Runs 1-6.

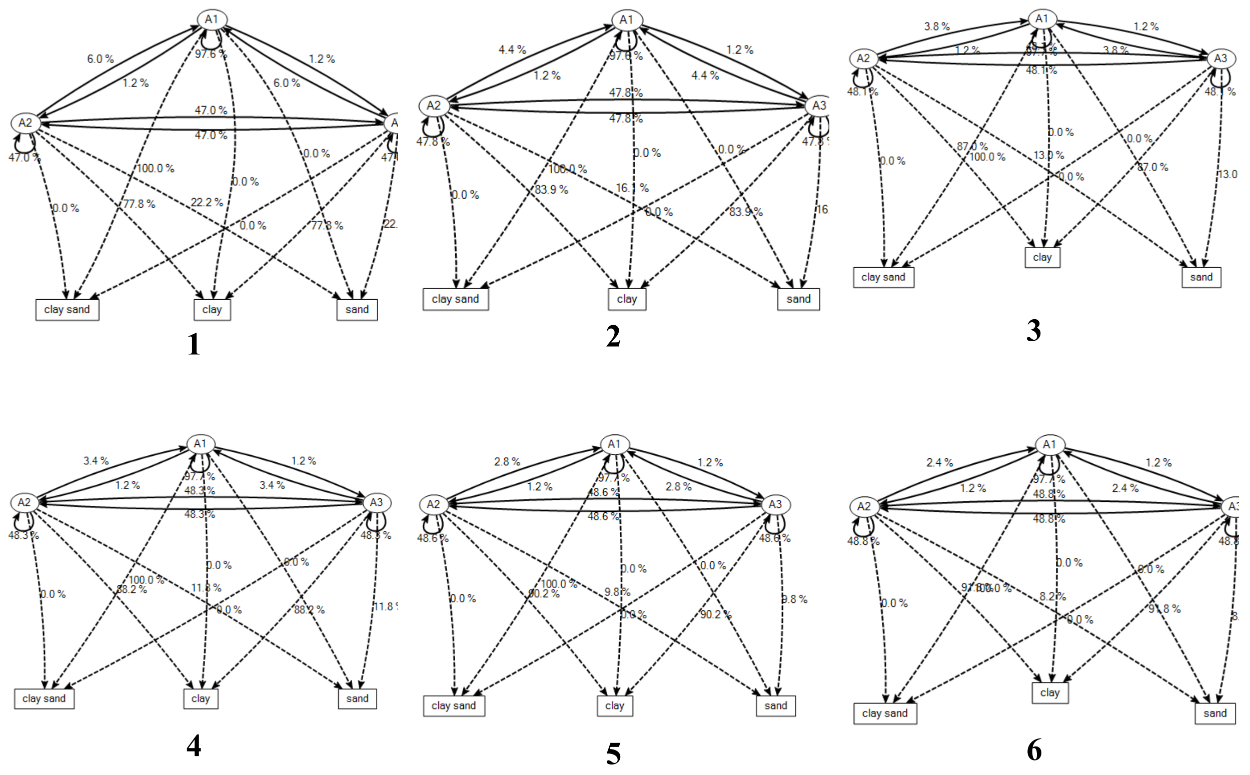


Figure 6.4: Evolution of the HMM Chain for Runs 1-6

Beginning with Run 1, hidden state A1 dominates the chain with a very low likelihood of transitioning to either A2 or A3. As is typically seen in HMM chains, each hidden state is, generally, more likely to be associated with a specific observation, which, in this case, is a certain ground condition. For example, A1 is generally associated with clay sand and has a self-transition probability of 97.6%. This approaches what is known as an absorbing state, which dominates the HMM chain. As the HMM is updated, the likelihood of observing clay increases for all hidden states. This behavior is expected as the previous project data was predominantly clay sand, whereas the Project B data used to update the HMM was most clay, especially through the first 200m of excavation. Sand was not seen at all during Project B excavation; this is also reflected in the evolution of the HMM with decreasing likelihoods from all states. The graphs in Figure 6.5 show a visual representation of what ground conditions were encountered in each simulation run.

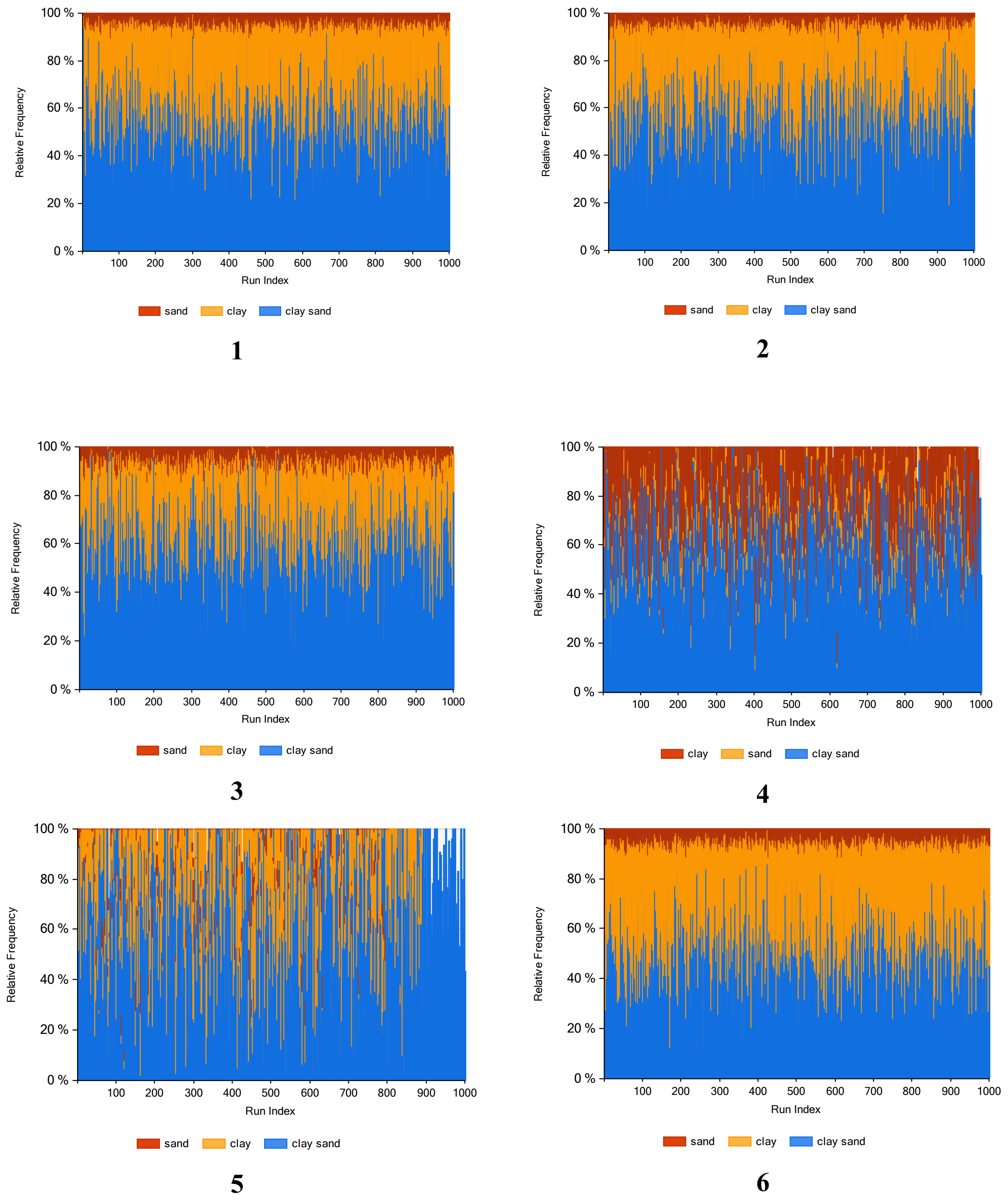


Figure 6.5: Ground conditions encountered for Runs 1-6

In Figure 6.5, ground conditions that were encountered can be determined in each run. Runs 1, 2, and 3 are still dominated by Project A data, where clay sand is the most encountered ground condition. In runs 4 and 5, actual construction data for Project B begins influencing the model.

The effects of the updating can be seen as the reduction in the number of observations of sand in runs 4 and 5. Run 6, for 500m of tunnel, yields the expected results: a higher and ;lower frequency of clay and sand, respectively, compared to Project A only data.

The project durations for each run are summarized in Table 6.5, with additional histograms and statistics provided in Appendix 6.B. It is important to note that each sequential run is 100m less than the previous, as the total project duration is based on the actual duration for data up to that point plus the simulation of the remaining chainage to 500m.

Table 6.5: Summary of run durations

Run Number	Simulated Mean Duration (hours)	Actual Duration (hours)	Total Duration (hours)
Run #1 (500m)	1491	0	**1491**
Run #2 (400m)	1185	211	**1396**
Run #3 (300m)	790	420	**1210**
Run #4 (200m)	527	652	**1179**
Run #5 (100m)	264	882	**1146**
Run #6 (500m)	1331	0	**1331**

Actual excavation duration for Project B was 1152 hours or 144 days of construction. This serves as the baseline comparison to evaluate the effectiveness of the updating procedures and completed simulation modelling. The results above indicate that performance vastly improved after each 100m of actual data was used to update the model. Beginning with Run #1, the predicted mean duration was 1491 hours or 186 days of construction. This estimate, based only on prior project data from Project A, serves as reasonable starting point for prediction. With Run #2, the prediction was reduced, based on 100m of actual data and ground condition information, to 1396 hours or 175 days. Run #3 resulted in a total estimated duration of 1210 work hours or 151 days. Runs #4, 1179 hours or 147 days, and Run #5, 1146 hours or 143 days, with information from 300m and 400m of actual data, respectively, began to accurately predict the actual project duration. These two runs in particular demonstrate that the simulation model and updating processes proposed resulted in marked improvements for predicting the duration of the project, which improved with the incorporation of additional data as expected. Achieving these reasonable results provides confidence that the methods used for updating provide a significant advantage that can enhance the planning and execution of a project.

The final completed Run #6 improves the initial estimate with only Project A data (Run #1). While the predicted duration of 1331 hours or 166 days was within the range of actual construction duration (144 days), these results demonstrate that the model is not perfectly calibrated to mirror the specific project. Notably, this is not the goal of the model. The important conclusion to draw from Run #6 is the marked improvement and convergence towards the actual duration in comparison to Run #1. Improvements in prediction, using the updating and Markov techniques, were shown to considerably improve the reliability of project duration predictions.

6.4 Conclusion

Being able to predict when a project will be completed is fundamental to project management in construction. The more accurate a project manager can be in predicting when a project will be completed, the better they can plan resources and materials to inform stakeholders and budgets.

A case study was reviewed to test the effectiveness and validity of proposed techniques in this thesis related to Bayesian updating of statistical distributions and Markov chains, specifically the Baum-Welch algorithm, to an actual construction project. The Downtown Intensification tunneling project in the city of Edmonton, Alberta, was used. Data was collected on a daily basis with respect to ground conditions encountered and production rates. Project A data were organized to develop baseline excavation rates for each ground condition as well as to train the HMM model based on the ground conditions and their persistence. A simulation was completed in the Simphony modelling environment with only the Project A data as an estimate for the 500m Project B. Four additional runs were completed to mimic the actual construction of 500m, and subsequent updates to the model were made at 100m increments. Each run was compared to the actual duration of the project. Each subsequent run demonstrated increasingly meaningful improvements, consecutively converging towards the actual duration. Visualizations were provided to display how the HMM model evolved through each update to demonstrate a visual summary of the predicted ground conditions.

The model runs demonstrate considerable improvements for each cycle of updates. Indeed, prediction, based on 400m of data, of the final actual duration was within one day of the actual duration. With further calibration and additional data analysis, the prediction could be improved as to converge towards the actual duration with less data, resulting in reduced uncertainty earlier in the project.

CHAPTER SEVEN

CONCLUSIONS

7.1 Summary of Thesis and Contributions

This thesis developed improvements in model-based predictions for construction projects with the use of Bayesian updating techniques and Markov chain modelling with integration into simulation modelling, specifically applied to geological conditions in tunneling. The background of the above techniques was reviewed with a summary of their origins, previous work done in construction, and a brief overview of how they function in Chapters 1 and 2.

In the third chapter, a model was proposed to apply Bayes technique to the updating of probabilistic transitions and observations for Markov chains. The effects of new data on predictions of ground conditions were visualized. It was shown that, based on a real sanitary sewer project, geological conditions of one-meter sections in tunnel construction can be predicted based on the geological condition of the previous one-meter section. The simulation results show that continuous updates during construction can improve project performance prediction by eliminating the uncertainty intrinsic to the original assumption.

The fourth chapter reviewed a specific algorithm for updating parameters of a Markov chain under the Baum-Welch algorithm—an applied Bayes technique. This algorithm was detailed and implemented into the Simphony modelling environment and was found to be accurate in recreating the results of previous works in Markovian modelling.

Chapter five of this thesis detailed the implementation of hidden Markov models (HMM) in the Simphony modelling environment with dedicated modelling elements for training, updating of model parameters, and integration of HMM models into a simulation model. A sample tunnel project was prepared with an integration of tunneling simulation together with the prediction of ground conditions via the Markov model.

The sixth chapter presented a real tunneling case study demonstrating the effectiveness of the integration of Bayesian updating techniques and hidden Markov models implemented in the Simphony simulation modelling environment that was applied to ground conditions for predicting project durations. Real construction data from a previous phase of a project was used to train an HMM. Actual data from the case study project was incrementally incorporated to update the HMM model, and Bayes techniques were applied to update the statistical distribution for excavation rates for the 500m project chainage. The results demonstrated that predictions were improved as additional, actual construction data were incorporated. This chapter demonstrated how project planners can make use of previous projects data as well as current, actual data to improve simulation model predictions.

Through the integration of well-established analytical techniques, this thesis presented how construction management and project planning can be improved by the innovative use of real construction data. Specifically applied to tunneling construction, the uncertainty of ground conditions was concentrated on to attempt to better model how ground conditions can change and how to better predict what conditions, and associated productivity, can be expected. The value of more accurate insight into a tunneling projects productivity is apparent and paramount in planning a project. Costs savings, stakeholder satisfaction, and better resource and equipment planning are all expected consequences of further understanding productivity. The straightforward implementation of the suggested techniques into models provides great potential for not only tunneling construction but for many other fields of construction where there is uncertainty surrounding production and the variables that drive it.

7.2 Recommendations for Future Work

To further develop the work in this thesis, a few key topics could be investigated. Additional model calibration for specific projects could be done to further improve the predictions and simulation of tunneling construction. Individual projects have various configuration and setups, and time studies into each facet of tunneling construction, such as other delays or activities in the process, could produce meaningful information. As a key driving force of productivity, ground conditions were the focus of Markovian modelling in this thesis, but modellers could expand the scope of Markov modelling to include other variables in construction, such as weather or crew size.

Further research could be done investigating how to weigh past and current project data to best model future performance of the project. Another topic of further research would be to investigate other techniques and algorithms for updating HMM parameters, outside of Bayes theorem and the

Baum Welch algorithm, to find the most accurate techniques. However, it should be considered that implementation into a modelling environment could be more complex for more robust algorithms.

In this thesis, the updating of statistical distributions was limited to normal distributions. In the future, it would be beneficial to expand updating to other types of statistical distributions that may better represent certain variables. There are certain limitations of sampling normal distributions, in particular, due to the range of the distributions that had to be explicitly prevented in this thesis, such as sampling of negative numbers.

CHAPTER 8

REFERENCES

Abdallah, M., Marzouk, M. (2013). Planning of tunneling projects using computer simulation and fuzzy decision making. *Journal of Civil Engineering And Management*, (4), 591. doi:10.3846/13923730.2013.793608

AbouRizk, S, Halpin, J. (December 1990). Probabilistic Simulation Studies for Repetitive Construction Processes. *Journal of Construction Engineering and Management.* Purdue University, West Lafayette, IN

AbouRizk, S., Mohamed, Y. (2000). Simphony: an integrated environment for construction simulation. *Paper presented at the Proceedings of the 32nd conference on Winter simulation.*

AbouRizk, S.M, Hajjar, Dany (1998) A framework for applying simulation in construction. *Canadian Journal of Civil Engineering*, 1998, 25(3): 604-617, 10.1139/l97-123

AbouRizk, S.M., Hague, S., Ekyalimpa, R. (2016) Construction Simulation: An Introduction Using SIMPHONY.

Al-Bataineh, M., AbouRizk, S., & Parkis, H. (2013). Using Simulation to Plan Tunnel Construction. *Journal Of Construction Engineering & Management*, *139*(5), 564-571. doi:10.1061/(ASCE)CO.1943-7862.0000626

Asadi, N., Mirzaei, A., Haghshenas, E. 2013. Multiple Observations HMM Learning by Aggregating Ensemble Models. *IEEE Transactions on Signal Processing.* 61(22): 5767-5776.

Balakrishnan, N, Nevzorov, V.B. (2004). A Primer on Statistical Distributions. Hoboken, New Jersey: John Wiley & Sons Inc Publication

Baum, L. E. (1972). An Inequality and Associated Maximization Technique in Statistical Estimation of Probabilistic Functions of a Markov Process. Inequalities, 1-8.

Baum, L. E., Eagon, J. A. (1967). An Inequality with Applications to Statistical Estimation for Probabilistic Functions of Markov Processes and to a Model for Ecology. *Bulletin of the American Mathematical Society,* 73(3), 360.

Baum, L. E., Petrie, T. (1966). Statistical Inference for Probabilistic Functions of Finite State Markov Chains. *The Annals of Mathematical Statistics,* 37(6), 1554-1563.

Baum, L. E., Sell, G. R. (1968). Growth Transformations for Functions on Manifolds. Pacific Journal of Mathematics, 27(2), 211-227.

Baum, L. E., Petrie, T., Soules, G., Weiss, N. (1970). A Maximization Technique Occuring in the Statistical Analysis of Probabilistic Functions of Markov Chains. *The Annals of Mathematical Statistics, 164.*

Bendat, Julius S, Piersol, Allan G (2011). Random Data: Analysis and Measurement Procedures (4th ed.) John Wiley & Sons Inc Publication

Bishop, C. M. (2006). Pattern Recognition and Machine Learning. (M. Jordan, J. Kleinberg, & B. Scholkopf, Eds.) New York, New York, USA: Springer Science + Business.

Bishop, T. M. (1986). Maximum Likelihood Alignment of DNA Sequences. *Journal of Molecular Biology,* 190(2), 159-165.

Bury, Karl (1999). Statistical Distributions in Engineering. Cambridge, UK: Cambridge University Press

Chung, T. H., Mohamed, Y., AbouRizk, S. (2006). Bayesian Updating Application into Simulation in the North Edmonton Sanitary Trunk Tunnel Project. *Journal Of Construction Engineering & Management*, *132*(8), 882-894. doi:10.1061/(ASCE)0733-9364(2006)132:8(882)

Dempster, A. P., Laird, N. M., Rubin, D. B. (1977). Maximum Likelihood from Incomplete Data Via the EM Algorithm. *Journal of the Royal Statistical Society,* 39(1), 1-38.

C. C. Gillespie. Dictionary of Scientific Biography, ed. (New York: Scribner's Sons, 1970), pp. 124–130.

Donghai, L., Peng, X., Shuai, L., Peizhi, H. (2015). Schedule Risk Analysis for TBM Tunneling Based on Adaptive CYCLONE Simulation in a Geologic Uncertainty-Aware Context. *Journal Of Computing In Civil Engineering*, *29*(6), 1-12. doi:10.1061/(ASCE)CP.1943-5487.0000441

Ebrahimy, Y., AbouRizk, S. M., Fernando, S., Mohamed, Y. (2011). Simulation modeling and sensitivity analysis of a tunneling construction project's supply chain. *Engineering Construction & Architectural Management (09699988)*, *18*(5), 462-480. doi:10.1108/09699981011074600

Eddy, S. R. (1995). Multiple Alignment Using Hidden Markov Models. In proceedings of the International Conference on Intelligent Systems in Molecular Biology. Vol. 3, pp. 114-120.

Einstein, H. (2004). Decision aids for tunneling: Update. *Transportation Research Record: Journal of the Transportation Research Board, 1892*(1), 199-207.

Farrokh, E., & Rostami, J. (2009). Effect of adverse geological condition on TBM operation in Ghomroud tunnel conveyance project. *Tunnelling And Underground Space Technology*, (4), 436.

Frazzoli, Emilio (2013). "Intro to Hidden Markov Models the Baum-Welch Algorithm" (PDF). *Aeronautics and Astronautics*, Massachusetts Institute of Technology.

Gang, C., Zhang-zhong, W., Fang-jie, W., You-li, M. (2011). Study on the application of a comprehensive technique for geological prediction in tunneling. *Environmental Earth Sciences*,*62*(8), 1667-1671. doi:10.1007/s12665-010-0651-y

Gaohui, W., Chao, W., Wenqi, S., Chengbo, D., Sherong, Z. (2014). Bayesian-based hybrid simulation approach to project completion forecasting for underground construction. *Journal Of Construction Engineering And Management*, (1)

Gilks, W. R. (2005). *Markov chain monte carlo*: Wiley Online Library.

Grinstead, C. M., Snell, J. L., (1997). Introduction to Probability. American Mathematical Society. ISBN 978-0-8218-0749-1.

Haas, C., & Einstein, H. H. (2002). Updating the decision aids for tunneling. *Journal of construction engineering and management, 128*(1), 40-48.

Hajjar, D., & AbouRizk, S. M. (2002). Unified Modeling Methodology for Construction Simulation. *Journal of Construction Engineering & Management*, *128*(2), 174.

Ines, Amor, V.M, Hansen. (2010), Enhancing the utility of daily GCM rainfall for crop yield prediction, *James W. Hansen and Andrew W. Robertson International Research Institute for Climate and Society,* The Earth Institute at Columbia University, Palisades, NY 10964-8000, USA

Ioannou, P. G. (1987). Geologic Prediction Model for Tunneling. *Journal Of Construction Engineering And Management*, *113*(4), 569.

Ioannou, P. G., & Martinez, J. C. (1996). Comparison of construction alternatives using matched simulation experiments. *Journal Of Construction Engineering & Management*, *122*(3), 231.

J. Banks; J. Carson; B. Nelson; D. Nicol (2001). *Discrete-Event System Simulation*. Prentice Hall. p. 3. ISBN 0-13-088702-1.

Jelinek, F., Bahl, L., & Mercer, R. (1975). Design of a Linguistic Statistical Decoder for the Recognition of Continuous Speech. *IEEE Transactions of Information Theory*, 21(3), 250-256.

Kemeny, J. G., Snell, J. L. (1976). Finite Markov Chains. *2nd Ed. New York Berlin Heidelberg Tokyo:* Springer-Verlag. p. 224. ISBN 978-0-387-90192-3.

Krogh, A., Brown, M, Mian, I. S., Sjoelander, K. (1994). Hidden Markov Models in Computational Biology: Applications to Protein Modeling. *Journal of Molecular Biology.* Vol. 235, Issue. 5, Pp. 1501.

Levinson, S. E., Rabiner, L. R., Sondhi, M. M. (1983). An Introduction to the Application of the Theory of Probabilistic functions of a Markov Process to Automatic Speech Recognition. *Bell System Technical Journal,* 62(4), 1035-1074.

Manabe, S., Hatanaka, T., Uosaki, K., Tabuchi, N., Matsuo, T., Hashizume, K. (2006). Training Hidden Markov Model Structure with Genetic Algorithm for Human Motion Pattern Classification. SICE-ICASE, 2006. *International Joint Conference* (pp. 618 - 622). Busan: IEEE.

Nefian, A., Hayes, M. (1998). Hidden Markov Models for Face Recognition. *In Proceedings of International Conference of Acoustics, Speech, and Signal Processing.* Seattle, WA, USA. Pp. 2721-2724.

Ng, C., Lu, H. (2014). Effects of the construction sequence of twin tunnels at different depths on an existing pile. *Canadian Geotechnical Journal*, (2), 173.

Norris, J. R. (1998). Markov chains. Cambridge University Press. Retrieved 2016-03-04. http://www.statslab.cam.ac.uk/~james/Markov/

Novak, D., Macas, M. (2008). Particle swarm optimization of hidden Markov models: A comparative study. *IEEE International Conference on Distributed Human-Machine Learning*. Athens: IEEE.

Petroff, V. (1989). Correction to: "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition".

Rabiner, L. R. (1989). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, 77(2):257-286.

Rabiner, L. R., Juang, B. H. (1986). An Introduction to Hidden Markov Models. *Institute of Electrical and Electronic Engineers (IEEE) ASSP Magazine.* Pp. 4-16.

Rabiner, L., Juang, B.-H. (1993). Fundamentals of Speech Recognition. Upper Saddle River, New Jersey, USA: Prentice Hall.

Rahimi, A. (2000). An Erratum for 'A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition.

Ranaweera, K., Ruwanpura, J., Fernando, S. (2013). Automated real-time monitoring system to measure shift production of tunnel construction projects. *Journal Of Computing In Civil Engineering*, (1), 68.

Richard Serfozo (24 January 2009). Basics of Applied Stochastic Processes. *Springer Science & Business Media.* p. 2. ISBN 978-3-540-89332-5.

Ritter, S., Einstein, H., Galler, R. (2013). Planning the handling of tunnel excavation material – A process of decision making under uncertainty. *Tunnelling And Underground Space Technology Incorporating Trenchless Technology Research,* 33193-201. doi:10.1016/j.tust.2012.08.009

R. L. Cave, L. P. Neuwirth (1980), Hidden Markov models for English, in J. D. Ferguson, editor, Hidden Markov Models for Speech, IDA-CRD, Princeton, NJ, October 1980.

Ruwanpura, J. Y., AbouRizk, S. M., Er, K., Fernando, S. (2001). Special purpose simulation templates for tunnel construction operations. *Canadian Journal of Civil Engineering,* 28(2), 222-237.

Shi, J., AbouRizk, S. M. (1997). Resource-based modeling for construction simulation. *Journal Of Construction Engineering & Management*, *123*(1), 26.

Sousa, R. L., Einstein, H. H. (2012). Risk analysis during tunnel construction using Bayesian Networks: Porto Metro case study. *Tunnelling And Underground Space Technology*, (1), 86.

Špačková, O., Straub, D. (2013). Dynamic Bayesian Network for Probabilistic Modeling of Tunnel Excavation Processes. *Computer-Aided Civil & Infrastructure Engineering*, *28*(1), 1-21. doi:10.1111/j.1467-8667.2012.00759.x

Stamp. Mark (2015), A Revealing Introduction to Hidden Markov Models. San Jose State University, Department of Computer Science.

Stigler, Stephen M. (1986). The History of Statistics: The Measurement of Uncertainty before 1900. *Cambridge, Massachusetts and London, England: The Belknap Press of Harvard University Press*

Stratonovich, R. L. (1960). Conditional Markov Processes. *Theory of Probability and its Applications,* 5(2), 156-178.

Tamir, Abraham (1998), Applications of Markov Chains in Chemical Engineering, ISBN: 978-0-444-82356-4. Elsevier Science.

Touran, A., Asai, T., Touran, A., Asai, T. (1987). Simulation of Tunneling Operations. *Journal Of Construction Engineering And Management*, *113*(4), 554.

Von Hilgers, Phillip, Langville, Amy N. (2003), The Five Greatest Applications of Markov Chains, College of Charleston. Charleston, SC 29424 USA.

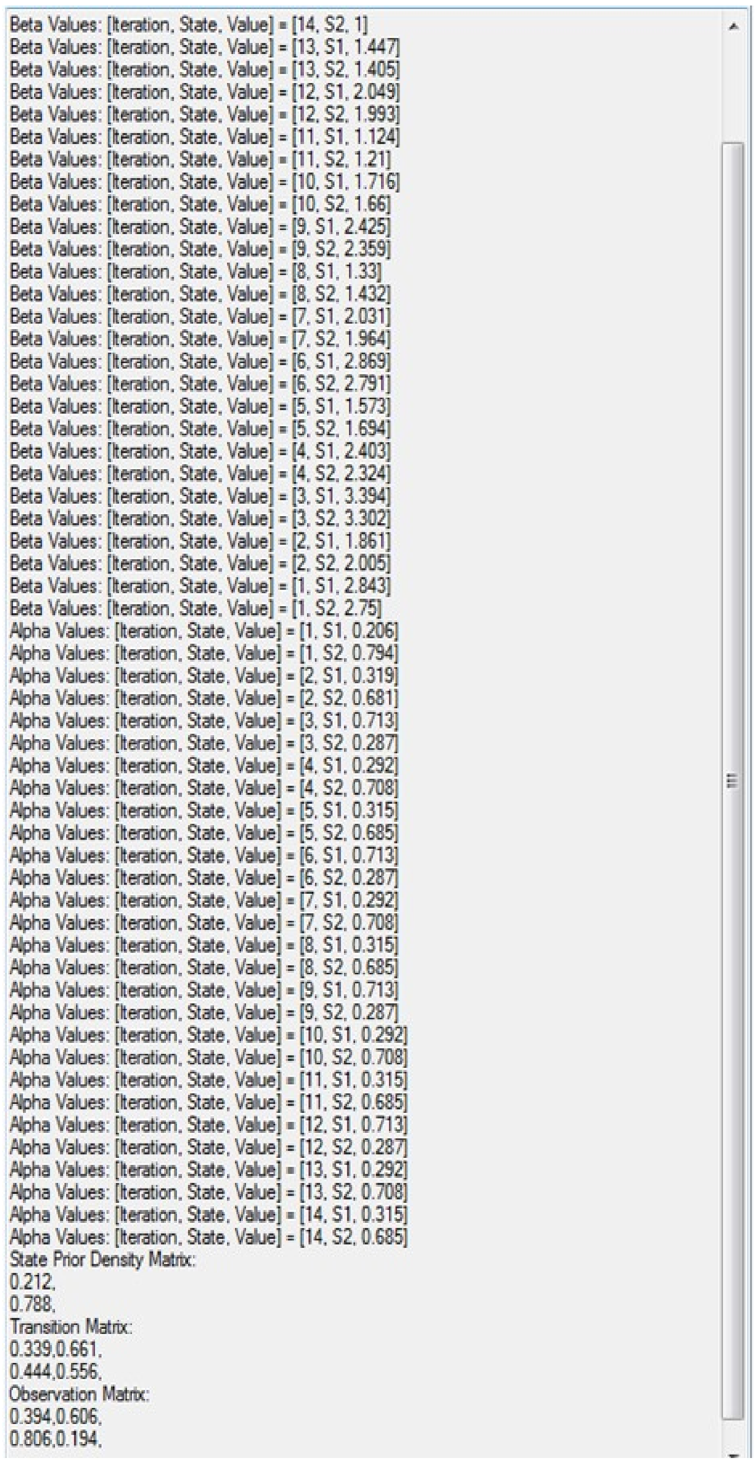
Wang, F., Ding, L., Love, P. E., Edwards, D. J. (2016). Modeling tunnel construction risk dynamics: Addressing the production versus protection problem. *Safety Science*, doi:10.1016/j.ssci.2016.01.014

Wilson, A., Bobick, A. (2001). Hidden Markov Chain Models for Modeling and Recognizing Gesture Under Variation. *International Journal for Pattern Recognition and Artificial Intelligence.* 15(1). Pp. 123-160.

Zhang, L., Wu, X., Skibniewski, M. J., Zhong, J., Lu, Y. (2014). Bayesian-network-based safety risk analysis in construction projects. *Reliability Engineering & System Safety, 131*(0), 29-39. doi: http://dx.doi.org/10.1016/j.ress.2014.06.006

APPENDICES

APPENDIX 4.A: Tracelog of Results from the Bayesian Updating Procedure in Simphony GPT:



```
Beta Values: [Iteration, State, Value] = [14, S2, 1]
Beta Values: [Iteration, State, Value] = [13, S1, 1.447]
Beta Values: [Iteration, State, Value] = [13, S2, 1.405]
Beta Values: [Iteration, State, Value] = [12, S1, 2.049]
Beta Values: [Iteration, State, Value] = [12, S2, 1.993]
Beta Values: [Iteration, State, Value] = [11, S1, 1.124]
Beta Values: [Iteration, State, Value] = [11, S2, 1.21]
Beta Values: [Iteration, State, Value] = [10, S1, 1.716]
Beta Values: [Iteration, State, Value] = [10, S2, 1.66]
Beta Values: [Iteration, State, Value] = [9, S1, 2.425]
Beta Values: [Iteration, State, Value] = [9, S2, 2.359]
Beta Values: [Iteration, State, Value] = [8, S1, 1.33]
Beta Values: [Iteration, State, Value] = [8, S2, 1.432]
Beta Values: [Iteration, State, Value] = [7, S1, 2.031]
Beta Values: [Iteration, State, Value] = [7, S2, 1.964]
Beta Values: [Iteration, State, Value] = [6, S1, 2.869]
Beta Values: [Iteration, State, Value] = [6, S2, 2.791]
Beta Values: [Iteration, State, Value] = [5, S1, 1.573]
Beta Values: [Iteration, State, Value] = [5, S2, 1.694]
Beta Values: [Iteration, State, Value] = [4, S1, 2.403]
Beta Values: [Iteration, State, Value] = [4, S2, 2.324]
Beta Values: [Iteration, State, Value] = [3, S1, 3.394]
Beta Values: [Iteration, State, Value] = [3, S2, 3.302]
Beta Values: [Iteration, State, Value] = [2, S1, 1.861]
Beta Values: [Iteration, State, Value] = [2, S2, 2.005]
Beta Values: [Iteration, State, Value] = [1, S1, 2.843]
Beta Values: [Iteration, State, Value] = [1, S2, 2.75]
Alpha Values: [Iteration, State, Value] = [1, S1, 0.206]
Alpha Values: [Iteration, State, Value] = [1, S2, 0.794]
Alpha Values: [Iteration, State, Value] = [2, S1, 0.319]
Alpha Values: [Iteration, State, Value] = [2, S2, 0.681]
Alpha Values: [Iteration, State, Value] = [3, S1, 0.713]
Alpha Values: [Iteration, State, Value] = [3, S2, 0.287]
Alpha Values: [Iteration, State, Value] = [4, S1, 0.292]
Alpha Values: [Iteration, State, Value] = [4, S2, 0.708]
Alpha Values: [Iteration, State, Value] = [5, S1, 0.315]
Alpha Values: [Iteration, State, Value] = [5, S2, 0.685]
Alpha Values: [Iteration, State, Value] = [6, S1, 0.713]
Alpha Values: [Iteration, State, Value] = [6, S2, 0.287]
Alpha Values: [Iteration, State, Value] = [7, S1, 0.292]
Alpha Values: [Iteration, State, Value] = [7, S2, 0.708]
Alpha Values: [Iteration, State, Value] = [8, S1, 0.315]
Alpha Values: [Iteration, State, Value] = [8, S2, 0.685]
Alpha Values: [Iteration, State, Value] = [9, S1, 0.713]
Alpha Values: [Iteration, State, Value] = [9, S2, 0.287]
Alpha Values: [Iteration, State, Value] = [10, S1, 0.292]
Alpha Values: [Iteration, State, Value] = [10, S2, 0.708]
Alpha Values: [Iteration, State, Value] = [11, S1, 0.315]
Alpha Values: [Iteration, State, Value] = [11, S2, 0.685]
Alpha Values: [Iteration, State, Value] = [12, S1, 0.713]
Alpha Values: [Iteration, State, Value] = [12, S2, 0.287]
Alpha Values: [Iteration, State, Value] = [13, S1, 0.292]
Alpha Values: [Iteration, State, Value] = [13, S2, 0.708]
Alpha Values: [Iteration, State, Value] = [14, S1, 0.315]
Alpha Values: [Iteration, State, Value] = [14, S2, 0.685]
State Prior Density Matrix:
0.212,
0.788,
Transition Matrix:
0.339,0.661,
0.444,0.556,
Observation Matrix:
0.394,0.606,
0.806,0.194,
```

APPENDIX 4.B: C# Code Snippet within the “Execute” Element in the Simphony for Updating HMM

```
//State Matrix
Matrix StateMatrix = new Matrix(2,1);
StateMatrix[0,0] = 0.5;
StateMatrix[1,0] = 0.5;
//Transition Matrix
Matrix TransitionMatrix = new Matrix(2,2);
TransitionMatrix[0,0] = 0.5;
TransitionMatrix[0,1] = 0.5;
TransitionMatrix[1,0] = 0.5;
TransitionMatrix[1,1] = 0.5;
//Observation Matrix
Matrix ObservationMatrix = new Matrix(2,2);
ObservationMatrix[0,0] = 0.3;
ObservationMatrix[0,1] = 0.7;
ObservationMatrix[1,0] = 0.7;
ObservationMatrix[1,1] = 0.3;
//List of all possible Observable states
List<string> L0 = new List<string>();
L0.Add("N");
L0.Add("E");
//List of all possible Hidden states
List<string> L1 = new List<string>();
L1.Add("S1");
L1.Add("S2");
//Observations Made
List<string> L2 = new List<string>();
L2.Add("N");
L2.Add("N");
L2.Add("E");
L2.Add("N");
L2.Add("N");
L2.Add("E");
L2.Add("N");
L2.Add("N");
L2.Add("E");
L2.Add("N");
L2.Add("N");
L2.Add("E");
L2.Add("N");
L2.Add("N");
//Create the HMM
HMMWithBayesianUpdating.HMM HMM = new
HMMWithBayesianUpdating.HMM(L1,L0,TransitionMatrix,ObservationMatrix,StateMat
rix);
HMM.StoreObservations(L2);
HMM.UpdateAllMarkovParameters();

TraceLine("************************************************************");
TraceLine("BW converged after " +
HMM.NumberOfIterationsTakenForParametersToConverge.ToString() + "
iterations");
TraceLine("************************************************************");
```

```
for(int i=0; i<HMM.ListOfBetaValues.Count; ++i)
{
  TraceLine("Beta Values: [Iteration, State, Value] = [" +
  HMM.ListOfBetaValues[i].Index.ToString() + ", " +
  HMM.ListOfBetaValues[i].State.ToString() + ", " +
  HMM.ListOfBetaValues[i].NormalizedValue.ToString() + "]");
}
TraceLine("");

for(int i=0; i<HMM.ListOfAlphaValues.Count; ++i)
{
  TraceLine("Alpha Values: [Iteration, State, Value] = [" +
  HMM.ListOfAlphaValues[i].Index.ToString() + ", " +
  HMM.ListOfAlphaValues[i].State.ToString() + ", " +
  HMM.ListOfAlphaValues[i].NormalizedValue.ToString() + "]");
}
TraceLine("");
TraceLine("");
for(int i=0; i<HMM.ListOfGammaValues.Count; ++i)
{
  TraceLine("Gamma Values: [Iteration, State, Value] = [" +
  HMM.ListOfGammaValues[i].Index.ToString() + ", " +
  HMM.ListOfGammaValues[i].State.ToString() + ", " +
  HMM.ListOfGammaValues[i].Value.ToString() + "]");
}

TraceLine("");
for(int i=0; i<HMM.ListOfZetaValues.Count; ++i)
{
  TraceLine("Zeta Values: [Iteration, Current State, Next State, Value] =
  [" + HMM.ListOfZetaValues[i].Index.ToString() + ", " +
  HMM.ListOfZetaValues[i].CurrentState.ToString() + ", " +
  HMM.ListOfZetaValues[i].NextState.ToString() + ", " +
  HMM.ListOfZetaValues[i].Value.ToString() + "]");
}

TraceLine("");
TraceLine("State Prior Density Matrix:");
for(int i=0; i<HMM.StateProbabilityMatrix_Current.RowCount; ++i)
{
  for(int j=0; j<HMM.StateProbabilityMatrix_Current.ColumnCount; ++j)
  {
      Trace(HMM.StateProbabilityMatrix_Current[i,j].ToString() + ",");
  }
  TraceLine("");
}
TraceLine("");
TraceLine("Transition Matrix:");
for(int i=0; i<HMM.TransitionMatrix_Current.RowCount; ++i)
{
  for(int j=0; j<HMM.TransitionMatrix_Current.ColumnCount; ++j)
  {
    Trace(HMM.TransitionMatrix_Current[i,j].ToString() + ",");
  }
  TraceLine("");
}
```

```
TraceLine("");
TraceLine("Observation Matrix:");
for(int i=0; i<HMM.ObservationMatrix_Current.RowCount; ++i)
{
   for(int j=0; j<HMM.ObservationMatrix_Current.ColumnCount; ++j)
   {
      Trace(HMM.ObservationMatrix_Current[i,j].ToString() + ",");
   }
   TraceLine("");
}

return true;
```

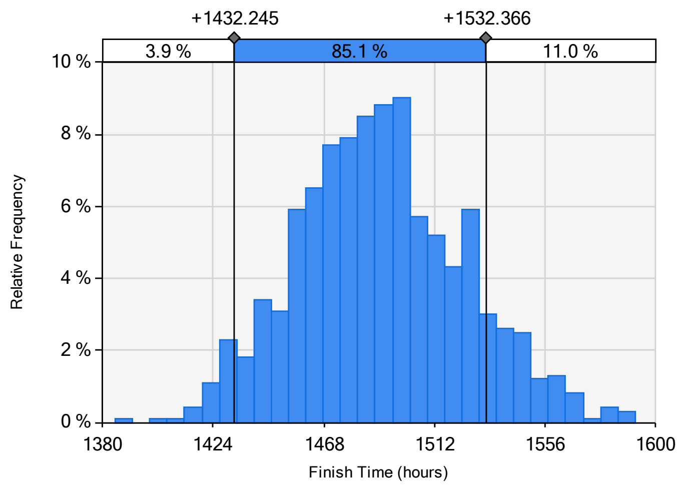
APPENDIX 6.A: Excavation Element Code

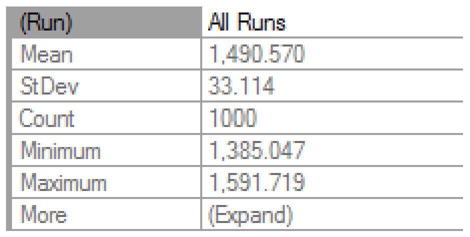
Excavate element code:

```
Dim X As Double

    Do
        If GetMarkovOutput("Ground Condition HMM")="clay"
            X = SampleNormal(2.34374827466158, 0.0827868300285651)
        Else If GetMarkovOutput("Ground Condition HMM")="clay sand"
            X = SampleNormal(2.34962480328052, 0.13553242570418)
        Else If GetMarkovOutput("Ground Condition HMM")="sand"
            X = SampleNormal(2.340,2.070)
        Else
            Throw New NotSupportedException("Unknown Ground Condition!")
        End If

    Loop While X < 1

    Return X
```

APPENDIX 6.B: Simulation Run Histograms

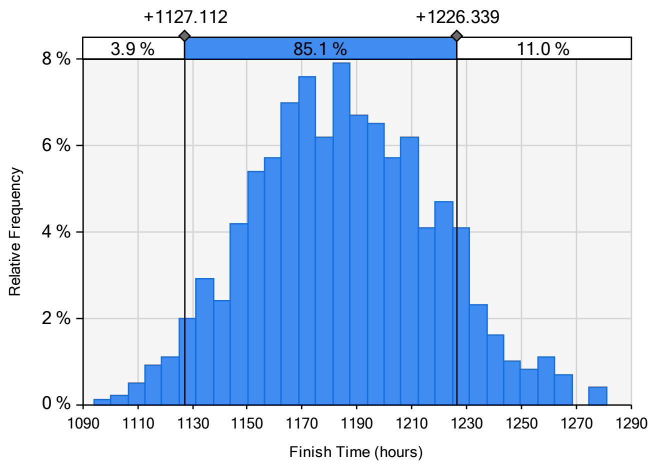
Run 1- 500m with Project A Trained Data

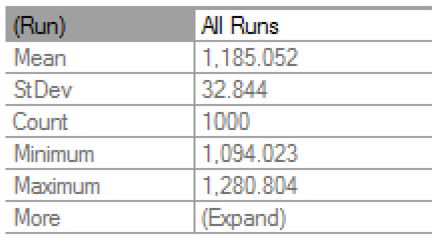




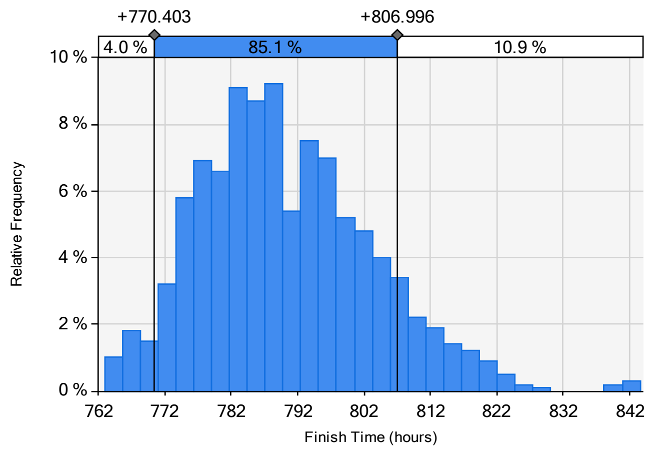
(Run)	All Runs
Mean	1,490.570
StDev	33.114
Count	1000
Minimum	1,385.047
Maximum	1,591.719
More	(Expand)

Run 2- 400m Updated with Project B-100m Actual Data





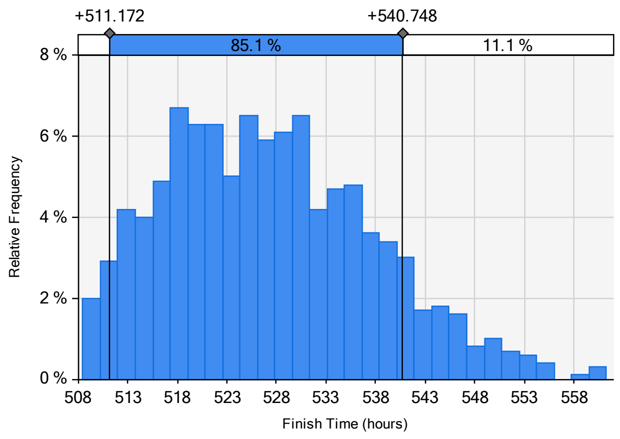
(Run)	All Runs
Mean	1,185.052
StDev	32.844
Count	1000
Minimum	1,094.023
Maximum	1,280.804
More	(Expand)

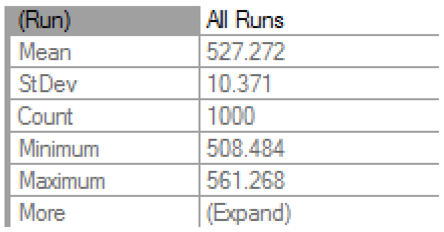


Run 3- 300m Updated with Project B-200m Actual Data

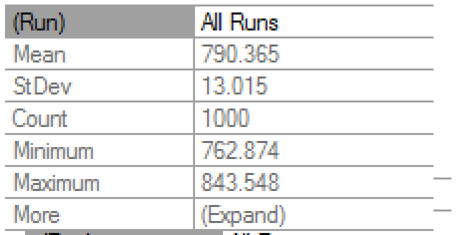
Run 4- 200m Updated with Project B-300m Actual Data

Run 5- 100m Updated with Project B-400m Actual Data

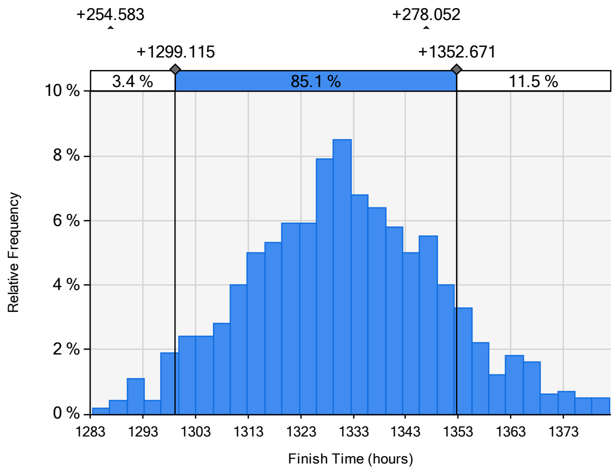


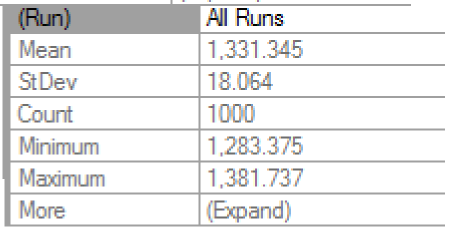


(Run)	All Runs
Mean	527.272
StDev	10.371
Count	1000
Minimum	508.484
Maximum	561.268
More	(Expand)



(Run)	All Runs
Mean	790.365
StDev	13.015
Count	1000
Minimum	762.874
Maximum	843.548
More	(Expand)



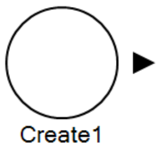


(Run)	All Runs
Mean	1,331.345
StDev	18.064
Count	1000
Minimum	1,283.375
Maximum	1,381.737
More	(Expand)

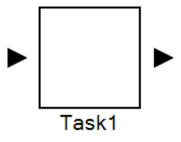
Run 6- 500m Updated with Project B compared to Actual Duration

APPENDIX 6.C: Simphony Modeling Elements

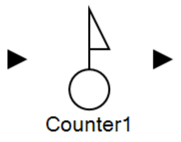
The following details of the SIMPHONY Modelling elements provided are excerpts Construction Simulation-An Introduction Using SIMPHONY.



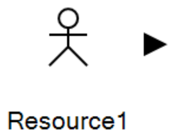
A Create element is responsible for creating entities and introducing them into the model. It has the capability of introducing a number of entities all at once at the start of simulation, or it can introduce single entities at random intervals during simulation.



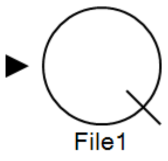
A Task element is responsible for modelling an activity. It achieves this by holding the entity for a period of time, as specified in its duration property.



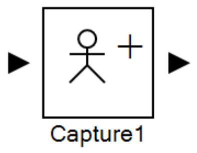
A Counter element is used to record important milestones in the lifecycle of an entity. It simply counts the number of entities passing though and the simulation time at which they are observed. Normally, it is used to record production in the model and should be strategically positioned to reflect the completion of a unit of production.



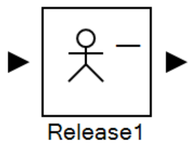
A Resource element is responsible for defining a shared resource. Although it has an output point, entities do not flow out of it. Rather, the output point is used to connect the Resource to one or more File elements. These File elements define the queues in which entities may be waiting for the Resource.



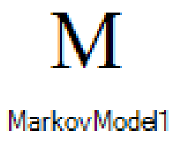
A File element is responsible for defining a queue in which entities wait for a shared resource. Although it has an input point, entities do not flow into it. Rather, the input point is used to connect the File to one or more Resource elements. These Resource elements define the resources that queued entities may be waiting for.



The Capture element is responsible for granting the exclusive use of one or more servers of a Resource to an entity. When an entity arrives at a Capture element, a check is made to see if the servers it requires are available. If so, the servers are granted to the entity and it exits the Capture element without delay; if not, the entity will be queued in a File element where it will wait for the servers to become available.



The Release element allows an entity to return servers it has previously captured to the pool of available servers.



The Markov Model element defines the Markov Chain and holds state and output information. No entities flow through the element, and it provides a user interface for the modeler to create a Markov Chain.

The Markov Transition element prompts a transition of the Markov Chain associated with it as well as an output. One transition and output will occur when an entity flows through the element.