# EMPIRICALLY DERIVED METHODS FOR ANALYSING SIMULATION MODEL OUTPUT

ALICIA MEJIA

The London School of Economics

Thesis submitted in fulfillment of the requirement for
the degree of Doctor of Philosophy at
the London School of Economics
University of London
September, 1992

## ABSTRACT

Often in simulation procedures are not proposed unless they are supported by a strong mathematical background. As will be shown in this thesis, this approach does not always give good results when the procedures are applied to complex simulation models, especially on output analysis. For this reason we have used an empirical rather than a theoretical approach for dealing with some of the output problems of simulation.

The research carried out has dealt mainly with queuing networks. The first problem we address is that of the identification of possible unstable queues. We also deal with the problem of the identification of queues that may require a long simulation run length to reach the steady state.

The method of replications is used for the estimation of terminating and sometimes of steady state parameters. In this thesis we study the relationship that exists between the number of replications used in the simulation and the simulation run length required for the parameter being estimated to reach the steady state. We also study the influence of the random number streams on the values of the mean estimates as a function of the number of replications.

One of the most commonly discussed problems related to the estimation of steady state parameters is that of the initialisation bias problem. Two methods are proposed in this thesis to deal with this problem. In one of the methods we propose an effective procedure that can be used for the estimation of the number of initial observations that are to be deleted. The second method, is based on a basic forecasting technique called weighted averages and does not require the elimination of any of the initial observations.

Another topic that has been studied in this thesis is the batch means method which is employed for the estimation of steady state parameters based on a single but very long simulation run. We show how a new sampling method called Descriptive Sampling is well suited for the estimation of steady state parameters with the batch means method. We also show how some of the procedures proposed in the literature for use in the batch means method do not work well in simulation models for which no analytical answer exists.

The thesis demonstrates that empirically derived methods can be practically effective and could form future theoretical research.

# ACKNOWLEDGMENTS

MY SINCERE THANKS TO THE FOLLOWING PEOPLE:

My supervisor, Dr. Ray J. Paul who, with his guidance and constant supervision, made this work possible.

My beloved father, for his teachings throughout my life and his constant support and encouragement.

Mark, for his patience and understanding.

Bert and Dimitris, for their loyal friendship and for being always ready to listen.

Ruby, for her help with grammar corrections.

The staff of LSE, especially Dr. J. Liebenau, for their advice and help.

Mr. A. Kiselstein and the people of A.K. Systems, for their advice and assistance in matters concerning computer hardware and software.

Finally, my family and friends in Colombia, for their prayers and interest in my work.

To my mother's memory

# TABLE OF CONTENTS

CHAPTER 1 : INTRODUCTION .......... 13
1.1. BASIC SIMULATION CONCEPTS .......... 13
1.1.1. WHAT IS SIMULATION? .......... 14
1.1.2. SIMULATION OBJECTIVES : WHEN IS IT USED? .......... 15
1.1.3. TYPES OF SIMULATION .......... 16
1.1.4. ESTIMATION OF SIMULATION PARAMETERS .......... 19
1.2. SOME PROBLEMS IN SIMULATION .......... 22
1.2.1. TERMINATING SIMULATIONS .......... 22
1.2.2. NON-TERMINATING SIMULATION .......... 23
1.3. RESEARCH OBJECTIVES .......... 25
1.4. RESEARCH METHODS .......... 28
1.4.1. JOE'S THEOREM .......... 30
1.5. THESIS OUTLINE .......... 31
1.6. SUMMARY .......... 32

CHAPTER 2 : RESEARCH DISCUSSION .......... 33
2.1. INTRODUCTION .......... 33
2.1.1. CHAPTER OBJECTIVES .......... 33
2.1.2. CHAPTER OUTLINE .......... 34
2.2. WHY SHOULD WE USE SIMULATION MODELS FOR WHICH NO ANALYTICAL ANSWER CAN BE CALCULATED ? .......... 35
2.3. QUEUING NETWORKS .......... 36
2.3.1. STATES OF A QUEUING NETWORK. .......... 37
2.3.2. SOME PARAMETERS OF INTEREST IN A QUEUING NETWORK .......... 38
2.4. HOW TO IDENTIFY CRITICAL QUEUES .......... 38
2.4.1. QUEUES THAT MAY NEVER APPEAR TO REACH A STEADY STATE. .......... 39
2.4.2. QUEUES THAT MAY TAKE A RELATIVELY LONG TIME TO REACH A STEADY STATE .......... 46
2.4.3. SIMULATION OF THE PUB MODEL UNDER DIFFERENT CONDITIONS .......... 48
2.5. INITIALISATION BIAS PROBLEM .......... 53
2.5.1. OSCILLATORY APPROACH TO THE STEADY STATE .......... 56

2.5.2. SMALL CHANGE IN THE INITIAL STATE OF THE QUEUE .......... 56
2.6. FURTHER CONSIDERATIONS .......... 57
2.6.1. FURTHER DISCUSSION ON SMALL MEAN ESTIMATES .. 57
2.6.2. DETERMINATION OF THE STEADY STATE .......... 58
2.6.3. LARGE STANDARD DEVIATION AS COMPARED WITH THE SAMPLE MEAN .......... 59
2.7. CONCLUSIONS .......... 60

CHAPTER 3 : ESTIMATION OF THE NUMBER OF REPLICATIONS REQUIRED IN A PARTICULAR SIMULATION .......... 62
3.1. INTRODUCTION .......... 62
3.1.1. CHAPTER OBJECTIVES .......... 63
3.1.2. CHAPTER OUTLINE .......... 63
3.2. ESTIMATING A SINGLE MEAN : SAMPLE SIZE REQUIRED ...... 64
3.2.1. ESTIMATION OF THE NUMBER OF REPLICATIONS ..... 64
3.3. INFLUENCE OF THE NUMBER OF REPLICATIONS ON THE ACCURACY OF THE MEAN ESTIMATES OBTAINED FROM THE SIMULATION .......... 67
3.3.1. INFLUENCE OF THE RANDOM NUMBER STREAMS ON THE ESTIMATES OBTAINED FROM A SIMULATION ... 67
3.3.2. NUMBER OF REPLICATIONS AND SIMULATION RUN LENGTH REQUIRED TO OBTAIN A GOOD ESTIMATE OF THE STEADY STATE .......... 69
3.4. ESTIMATION OF THE NUMBER OF REPLICATIONS REQUIRED FOR THE ACCURATE ESTIMATION OF A PARAMETER IN SIMULATION .......... 71
3.4.1. PROPOSED METHOD FOR THE ESTIMATION OF THE NUMBER $K$ OF REPLICATIONS .......... 71
3.5. ANALYSIS OF THE EMPIRICAL RESULTS .......... 73
3.5.1. ESTIMATION OF THE NUMBER OF REPLICATIONS ..... 73
3.5.2. EVALUATION OF THE PERFORMANCE OF THE PROPOSED PROCEDURE. .......... 74
3.6. CONCLUSIONS .......... 80

CHAPTER 4 : STEADY STATE : REDUCING THE TRANSIENT PHASE ....... 82
4.1. INTRODUCTION .......... 82

4.1.1. CHAPTER OBJECTIVES ........................... 85
4.1.2. CHAPTER OUTLINE ............................. 85
4.2. TRANSIENT PHASE :WHAT IT IS ............................ 86
4.2.1. DELETION OF SOME OF THE INITIAL OBSERVATIONS DURING THE TRANSIENT PHASE ................. 87
4.3. SOME METHODS FOR THE ELIMINATION OF INITIALISATION BIAS IN SIMULATION ................................. 89
4.3.1. KELTON AND LAW'S METHOD ...................... 90
4.3.2. KELTON AND LAW'S METHOD: DESCRIPTION ......... 91
4.3.3. WELCH'S METHOD ............................... 96
4.4. A NEW METHOD FOR THE ELIMINATION OF THE INITIALISATION BIAS PROBLEM ......................... 98
4.4.1. PROPOSED NEW METHOD : ITS BASIS. ............... 99
4.4.2. GORDON'S AND CONWAY'S RULES .................. 100
4.4.3. USE OF THE STANDARD DEVIATION TO ESTIMATE A "GOOD" RUN-IN-PERIOD : AN ALGORITHM .......... 104
4.4.4. COMPARISON OF OUR METHOD AND GORDON'S METHOD FOR THE ESTIMATION OF A RUN-IN-PERIOD. ........................................ 109
4.5. EMPIRICAL RESULTS. ........................................ 112
4.5.1. MEASURES OF PERFORMANCE ..................... 112
4.5.2. ANALYSIS OF THE RESULTS FOR THE LAUNDERETTE AND THE MILITARY MODELS ..................... 113
4.6. SIMULATION OF SOME SYSTEMS FOR WHICH AN ANALYTICAL ANSWER EXISTS ..................... 121
4.6.2. ANALYSIS OF THE RESULTS OBTAINED FOR THE M/M/1 QUEUE ................................. 124
4.6. CONCLUSIONS ............................................ 128

CHAPTER 5 : CONFIDENCE INTERVALS IN STEADY STATE SIMULATION ... 131
5.1. INTRODUCTION ........................................... 131
5.1.1. CHAPTER OBJECTIVES ........................... 132
5.1.2. CHAPTER OUTLINE ............................. 133
5.2. THEORETICAL BACKGROUND ............................. 133
5.3. DESCRIPTIVE SAMPLING IMPLEMENTATION ................. 135
5.3.1. SET VALUES GENERATION. ........................ 136
5.3.2. RANDOM PERMUTATION ......................... 137

5.4. BATCH MEANS METHOD .................................. 138
5.4.1. USE OF A SINGLE LONG RUN FOR THE ESTIMATION OF STEADY STATE PARAMETERS : GENERAL DISCUSSION. .................................. 138
5.4.2. DESCRIPTION OF THE BATCH MEANS FOR STEADY STATE PARAMETERS ESTIMATION. ................ 140
5.4.3. SEQUENTIAL PROCEDURES ........................ 141
5.5. DISCUSSION OF THE BATCH MEANS METHOD. ............... 143
5.5.1. MODIFICATION OF THE BATCH MEANS METHOD ..... 144
5.5.2. POINTS RELATED TO THE APPLICATION OF THE BATCH MEANS METHOD TO COMPLEX SIMULATION MODELS .................................. 144
5.5.3. DESCRIPTION OF THE EXPERIMENTS ............... 145
5.5.4. ANALYSIS OF THE RESULTS OBTAINED FOR THE LAUNDERETTE MODEL. .......................... 146
5.5.5. SUMMARY OF OUR RESULTS CONCERNING THE USE OF THE BATCH MEANS METHOD .................. 153
5.6. ALGORITHM .......................................... 153
5.7. CONCLUSIONS .......................................... 156

CHAPTER 6 : ALTERNATIVE METHODS OF EVALUATING STEADY STATE PARAMETERS .......................................... 160
6.1. INTRODUCTION .......................................... 160
6.1.1. CHAPTER OBJECTIVES .......................... 161
6.1.2. CHAPTER OUTLINE .............................. 161
6.2. USE OF WEIGHTED AVERAGES TO EVALUATE STEADY STATE PARAMETERS. .......................................... 162
6.2.1. WEIGHTED AVERAGES .......................... 162
6.2.2. WEIGHTED AVERAGES IN STEADY STATE SIMULATION .................................. 162
6.3. EMPIRICAL RESULTS. .................................. 164
6.3.1. DESCRIPTION OF THE EXPERIMENTS ............... 165
6.3.2. ANALYSIS OF THE RESULTS FOR THE PUB MODEL .... 165
6.4. WEIGHTED AVERAGES METHOD APPLIED TO A SYSTEM WITH KNOWN ANALYTICAL ANSWER. .......................... 170
6.5. CONCLUSIONS .......................................... 172

CHAPTER 7 : CONCLUSIONS . . . . . 174
7.1. SUMMARY . . . . . 174
7.2. CONCLUSIONS . . . . . 175
7.3. FUTURE RESEARCH . . . . . 178
7.3.1. USE IN VERY COMPLEX SIMULATION MODELS . . . . . 178
7.3.2. DIFFERENT PARAMETERS . . . . . 179
7.3.3. IMPLEMENTATION OF DESCRIPTIVE SAMPLING . . . . . 179
7.3.4. SOFTWARE IMPLEMENTATION . . . . . 179
7.3.5. ALTERNATIVE STATISTICS: BAYESIAN AND NON-PARAMETRIC . . . . . 180
7.3.6. SIMULATION RUN LENGTH . . . . . 181
7.3.7. PROBLEM OF INITIALISATION BIAS: ALTERNATIVES. . . 181
7.3.8. PROPORTIONS RATHER THAN MEAN ESTIMATES. . . . . 182
7.3.9. OPTIMISATION . . . . . 182
7.4. FINAL WORDS . . . . . 183

APPENDICES . . . . . 184

APPENDIX A : ACTIVITY CYCLE DIAGRAMS . . . . . 185
A.1. INTRODUCTION . . . . . 185
A.2. SOME EXAMPLES . . . . . 186
A.2.1. The PUB . . . . . 187
A.2.2. The LAUNDERETTE Example . . . . . 192
A.2.3. The STEELWORKS example. . . . . . 194
A.2.4. THE FISH PACKING Example . . . . . 197
A.2.5. THE BRAZILIAN HOSPITAL Example . . . . . 199
A.2.6. THE MILITARY Example . . . . . 201
A.2.7. JACKSON'S MODEL . . . . . 203

APPENDIX B : ADDITIONAL EXAMPLES FOR CHAPTER 2 . . . . . 205
B.1. INTRODUCTION . . . . . 205
B.2. CRITICAL QUEUES: POSSIBLE REASONS FOR $\lambda/s\mu > 1$. . . . . . 205
B.3. QUEUES THAT MAY TAKE A RELATIVELY LONG TIME TO REACH THE STEADY STATE . . . . . 212
B.3.1. CRITICAL QUEUES DUE TO THE CHARACTERISTICS OF THE MODEL. . . . . . 212

B.3.2. CRITICAL QUEUES: INFLUENCE OF CHANGES IN THE CHARACTERISTICS OF THE MODEL ................ 216
B.3.3. ADDITIONAL CONSIDERATIONS .................... 219

APPENDIX C : STEADY STATE VALUES FOR SOME SIMULATION MODELS ................................................ 222
C.1. INTRODUCTION ........................................ 222
C.2. DISCUSSION ON THE VALUE OF $\epsilon$ AS AN APPROXIMATION FOR THE REAL VALUE $\mu$ WHICH IS BEING ESTIMATED...... 223
C.3. STEADY STATE ESTIMATES FOR THE VALUE $\mu$. .............. 223
C.3.1. THE LAUNDERETTE .............................. 223
C.3.2. THE PUB ........................................ 226
C.3.3. THE STEELWORKS. .............................. 229
C.3.4. THE MILITARY MODEL ........................... 231
C.3.5. THE FISH PACKING MODEL ......................... 233

APPENDIX D : ADDITIONAL RESULTS CORRESPONDING TO THE SIMULATION MODELS USED IN CHAPTER 3 .................... 235
D.1. INTRODUCTION ........................................ 235
D.2. ANALYSIS OF THE EMPIRICAL RESULTS OBTAINED FOR SOME SIMULATION MODELS ................................ 235
D.2.1. THE LAUNDERETTE MODEL : ESTIMATION OF THE NUMBER OF REPLICATIONS ...................... 236
D.2.2. THE LAUNDERETTE MODEL : EVALUATION OF THE PERFORMANCE OF THE PROPOSED PROCEDURE ... 236
D.2.3. THE FISH PACKING MODEL : ESTIMATION OF THE NUMBER OF REPLICATIONS ...................... 242
D.2.4. THE FISH PACKING MODEL : EVALUATION OF THE PERFORMANCE OF THE PROPOSED PROCEDURE ... 242
D.2.5. THE STEELWORKS MODEL : ESTIMATION OF THE NUMBER OF REPLICATIONS ...................... 248
D.2.6. THE STEELWORKS MODEL: EVALUATION OF THE PERFORMANCE OF THE PROPOSED PROCEDURE..... 248

APPENDIX E: GORDON'S METHOD FOR THE ELIMINATION OF THE INFLUENCE OF THE INITIAL CONDITIONS ..................... 250
E.1. GORDON'S METHOD : DESCRIPTION ...................... 250

E.2. DISCUSSION OF GORDON'S METHOD ........................ 251
E.3. EMPIRICAL RESULTS ........................................ 251
E.3.1. THE PUB ............................................... 252
E.3.2. THE LAUNDERETTE ....................................... 255
E.3.3. THE STEELWORKS ........................................ 255
E.3.4. THE M/M/1 QUEUE. ...................................... 258
E.3.5. THE M/M/4 QUEUE ....................................... 258
E.3.6. JACKSON'S MODEL ....................................... 260

APPENDIX F : FURTHER ANALYSIS OF THE EMPIRICAL RESULTS CORRESPONDING TO THE SIMULATION MODELS USED IN CHAPTER 4 ................................................ 262
F.1. INTRODUCTION ............................................. 262
F.2. MEASURES OF PERFORMANCE. ................................. 262
F.3. ANALYSIS OF THE RESULTS .................................. 263
F.3.1. THE LAUNDERETTE. ...................................... 263
F.3.2. THE PUB. .............................................. 274
F.3.3. THE STEELWORKS ........................................ 281
F.4. ANALYSIS OF SOME SYSTEMS WITH KNOWN ANALYTICAL STEADY STATE VALUE. ..................................... 288
F.4.1. M/M/4 QUEUE ........................................... 288
F.4.2. 2-STAGE QUEUING SYSTEM ................................. 293
F.5. CONCLUSIONS .............................................. 297

APPENDIX G : ADDITIONAL RESULTS FOR CHAPTER 5. ................. 299
G.1. INTRODUCTION ............................................. 299
G.2. STEADY STATE SIMULATION .................................. 299
G.2.1. DISCUSSION OF THE BATCH MEANS METHOD. ................. 299
G.2.2. MODIFICATION OF THE BATCH MEANS METHOD ................ 300
G.3. ANALYSIS OF THE RESULTS .................................. 300
G.3.1. DESCRIPTION OF THE EXPERIMENTS ........................ 301
G.3.2. ANALYSIS OF THE RESULTS. .............................. 301

APPENDIX H : TESTS FOR CORRELATION OF SIMULATION OUTPUT DATA ................................................... 311
H.1. INTRODUCTION ............................................. 311
H.2. TEST BASED ON THE VON NEUMANN STATISTIC. ................. 311

H.3. SOME OTHER TESTS FOR CORRELATION.................... 312

APPENDIX I : ADDITIONAL RESULTS FOR CHAPTER 6 .................. 314
I.1. INTRODUCTION .......................................... 314
I.1.1. WEIGHTED AVERAGES AND THEIR USE IN SIMULATION .................................. 314
I.1.2. MEASURES OF PERFORMANCE ..................... 315
I.2. EMPIRICAL RESULTS.......................................... 316
I.2.2. THE FISH PACKING MODEL......................... 321
I.2.3. THE STEELWORKS MODEL .......................... 325
I.2.4. The M/M/4 QUEUE ................................ 327
I.2.5. JACKSON'S MODEL ................................ 328

APPENDIX J : PASCAL MODIFICATIONS TO USE THE METHOD OF WEIGHTED AVERAGES ........................................ 332
J.1. INTRODUCTION .......................................... 332
J.2. QUEUING TIME PROCEDURE ............................. 332
J.3. QUEUE LENGTH PROCEDURE ............................ 334

REFERENCES ................................................ 337

BIBLIOGRAPHY .............................................. 356

# CHAPTER 1 : INTRODUCTION

## 1.1. BASIC SIMULATION CONCEPTS

In the last two or three decades there has been a fast and important development in computer technology. As a consequence of this, the world has been transformed. Computers have become smaller, cheaper and more powerful. They are not only a useful tool in today's world, but they are in many ways necessary; airline bookings are not possible when "the system is down"; bank transactions are equally affected when the computer is not working, just to give two examples of how much businesses rely on computers. Most universities, large and small, include in their curriculum at least one course on computers. But their use is not limited only to the business and academic worlds. Computers are found in households, charity centres, hospitals etc. And this popularity and availability of computers has been an important factor in the development and use of simulation. (For example, Crookes and Valentine (1982) describe a visual colour simulation development on APPLE computers carried out to assess an expensive capital installation; in Chapter 3, Law and Kelton (1991), give a description of different types of software that are currently used in simulation; see also Hollocks (1984)).

### 1.1.1. WHAT IS SIMULATION?

Several definitions of simulation can be found in the literature. A very basic definition is the following:

"Simulation is the construction of a mathematical model for some process, situation, etc, in order to estimate its characteristics or solve problems about it probabilistically in terms of the model." (The COLLINS English Dictionary, 1986).

A more formal definition of simulation is the following:

"Simulation is essentially a controlled statistical sampling technique (experiment) that is used, in conjunction with a model, to obtain approximate answers for questions about complex, multifactor probabilistic problems." (Lewis and Orav, 1989).

However, no matter how formal and complex a definition is, it is always possible to identify in all of them the main characteristics of *simulation*: a *model* of the system is developed, and from this model, and using *statistical and mathematical techniques*, some inferences about its behaviour may be drawn.

Although simulation could be performed without a computer, in practice the real-world systems that need to be analysed with the help of simulation are quite complex, and performing the simulation by hand would take a long time, would be tedious and would be a source of errors.

### 1.1.2. SIMULATION OBJECTIVES : WHEN IS IT USED?

Before starting any research in the area of simulation it is very important to understand why and when simulation is used. Basically, it may be said that "simulation is run in order to gain an understanding of the behaviour of the system under study." (Seila, 1990). A better understanding of the use of simulation can be obtained by mentioning a few of the fields where it has been successfully used:

1. Computer Systems.
2. Communication Systems.
3. Environmental and Energy Flow.
4. Crop Management and Ecological Studies.
5. Transportation Systems.
6. Policy Analysis.
7. Project Planning and Control. (See Pritsker, 1984).

Common to all these studies is the fact that the systems are too complex and, therefore an analytical answer to the problem(s) does not exist. This last point is very important: simulation is used to give an answer to probabilistic problems. Due to this *probability* aspect of simulation, there is some "uncertainty" which is inherent to the results obtained from simulation. Therefore if a mathematical technique exists that, without making great assumptions (hardly met in practice) can provide an exact answer, this technique should be used instead of the more sophisticated but in some ways "uncertain" technique of simulation.

The dangers of using simulation when it is not necessary are clearly illustrated in the following paragraph, which is part of a letter received by Woolsey (1979) as a response to his article : "Whatever Happened to Simple Simulation". The author of the letter "wants to remain anonymous, probably for reasons of national security." (Woolsey, 1979). In this letter the author refers to a problem that was tackled by simulation and "after careful analysis of old records which initiated the study in the first place (strangely enough, these were found among the belongings of Ramses II), I have concluded that simulation wasn't necessary, as application of simple analytic tools to subproblems would have provided excellent results. Anyway, the problem is no longer a problem as so much time has passed that the stuff that the computer model was to have predicted has happened without the termination of life on earth. Unfortunately, if I brought this fact to the attention of management I would certainly be locked up "down in the mine" for the rest of my born days."

Some suitable areas of application for simulation are then the following:

1. Analysis of complex systems.

2. Forecasting of possible effects of changes in the number of resources or their allocation.

3. Determination of the critical variables in a system and of the way they interact in the system.

4. To test a system before it is built because in some cases it may be very difficult or even impossible from a practical point of view: for example, in the simulation of a naval battle.

### 1.1.3. TYPES OF SIMULATION

There are several types of simulation models depending on the type of variables used (random or deterministic), length of the run (finite or infinite), type of change of the state variables (discrete or continuous) etc. Some of these types of simulation are defined in this subsection.

#### STATIC OR DYNAMIC SIMULATION

A simulation model is called *static* when it represents a system at a particular point of time. It is sometimes called a Monte Carlo simulation.

When the simulation model represents a system that changes over time it is called *dynamic* simulation.

#### DETERMINISTIC OR STOCHASTIC SIMULATION

In a *deterministic* simulation, the variables are exactly determined; this means that they are properly specified instead of being generated from a probability function. In a *stochastic* simulation one or more of the variables are random, i.e., they are defined according to a probability density function. For example, the arrival of customers in a bank or in a post office may not happen at regular intervals of time but might follow a random pattern which can be modelled by the exponential distribution. Identifying an adequate input distribution may not be easy but nevertheless if simulation is going to be successful this is a critical aspect of simulation modelling. Examples of research in this field can be found in Cochran and Cheng Chuen-Sheng (1990), DeBrota et al (1988), Avramidids and Wilson (1989), DeBrota et al (1989).

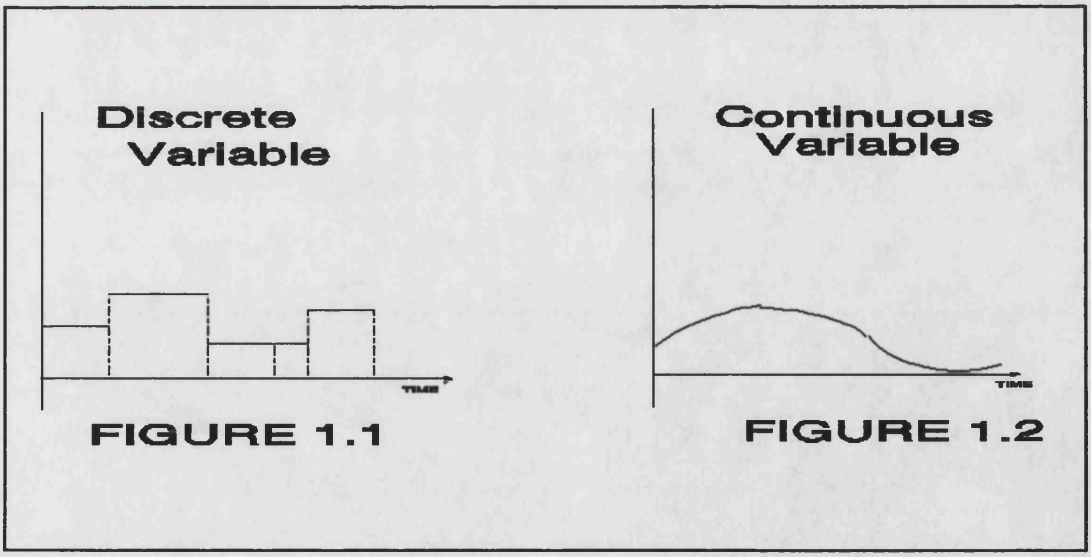
#### DISCRETE OR CONTINUOUS SIMULATION

Law and Kelton (1991) give the following definition:

"*Discrete Event Simulation* concerns the modelling of a system as it evolves over time, by a representation in which the state variables change instantaneously at separate points in time. (In more mathematical terms, we might say that the system can change at only a *countable* number of points in time.) These points in time are the ones at which an event occurs where an *event* is defined to be an instantaneous occurrence that may change the state of the system." In a simulation where customers arrive randomly (post office, banks, pubs, launderettes, etc.) the number of customers waiting at a particular queue to be served constitutes a state of the system. Therefore, the arrival of a new customer to a system is an event.

In a "*continuous simulation* the state variables change continuously over time. An example is the head of water behind a dam. During and for some time after a rain storm, water flows into the lake behind the dam. Water is drawn from the dam for flood control and to make electricity. Evaporation also decreases the water level." (Banks and Carson, 1984)

Figures 1.1. and 1.2. illustrate the difference between discrete and continuous state variables, and how they change in value over time.




**Figures 1.1 and 1.2.** Discrete and continuous system state variable.

## TERMINATING OR NON-TERMINATING SIMULATION

In some cases the simulation ends when an "special event" occurs. That is, "there is a natural event E which specifies the length of each run (replication)" (Law, 1990). For example, the simulation of a post office might consider the system for a period of 8.30 hours (9.00 a.m. to 5.30 p.m.). Similarly a bank may be simulated for some pre-specified period of time (say seven hours: from 9.30 a.m. to 4.30 p.m.). In this case the simulation is called *terminating*. In contrast with this type of simulation, we have *non-terminating* simulations. In this latter type, the simulation run length is not decided by the system that is being simulated, but it is a choice of the simulation practitioner. In other words, in contrast with terminating simulations, in non-terminating simulations there is no natural event E to specify the length of the run. Although in theory a non-terminating simulation is exactly that: non-terminating, most systems after being simulated for a reasonably long period of time tend to become "stable" or, in other words, reach a "steady state". To explain this, suppose that the simulation is estimating the waiting time for a customer in a system that operates day and night. Let us call $W_1, W_2, ...W_n$ the waiting time of customers 1, 2, ..n respectively. The "steady state" mean waiting time $\bar{W}$ is given by

$$\bar{W} = \lim_{n \to \infty} \frac{1}{n} \sum_{i=1}^{n} W_i \qquad (1.1)$$

This means that the mean waiting time is the waiting time for a very large (infinite) number of customers in the system. After a "long", but finite duration, the value of W does not change too much.

However, another way of interpreting "steady state" is by defining the steady state as that time "when the <u>distribution</u> of the parameter that is being estimated (in this case, the mean waiting time) becomes invariant" (Law and Kelton, 1982b). It is important to notice that it is the distribution, and not the actual values of the parameter being estimated, that becomes invariant. Another way of interpreting this definition of "steady state" is by saying that in the "steady state" the system is independent of the initial conditions. This means that the distribution of the waiting time is exactly the same, independent of the number of customers initially present in the system, and when the simulation was started. For example, whether the system has 5 or 100 customers at the start of the simulation, in the steady state the distribution of the waiting times will be exactly the same.

### 1.1.4. ESTIMATION OF SIMULATION PARAMETERS

Depending on the type of simulation, terminating or steady state, different methods are used for the estimation of parameters. We will describe in this sub-section some of the most common methods employed for this estimation, for these two different types of simulation.

**a. Terminating Simulations**

In terminating simulations parameters are normally estimated using the method of replications. In this method *n* different replications of the model are run, each one using different random number seeds in order to ensure independence of the observations. For each replication an estimate $X_i$ is obtained. If the parameter being estimated is a mean value $\bar{X}$, this will be easily estimated as

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i \tag{1.2}$$

One of the problems faced by the simulation practitioner is then that of deciding how many replications are required to obtain an accurate parameter.

**b. Steady state simulations**

Several methods have been proposed in the literature for the estimation of steady state parameters. Among these methods we can mention the following:

1. Replications method.
2. Batch means method.

3. Regenerative method.

The method of replications has already been described and in this case besides the problem of deciding on the number of replications to be used, the simulation practitioner faces the problem of how long the simulation run should be for the mean estimate to be independent of the initial conditions. This problem known as the problem of the initialisation bias is discussed more in detail in Section 1.2.2.

As opposed to the replications method, the batch means and the regenerative method obtain the steady state mean estimates *using a single but very long simulation run.*

**BATCH MEANS METHOD**

In the BATCH MEANS method (Fishman, 1976) N observations $X_1$, $X_2..X_N$ are recorded. These observations are grouped into b batches of size N/b and the mean $\bar{X}_i$ of each one of these batches is calculated as:

$$\bar{X}_i = \sum_{j=1}^{N/b} \frac{X_{j+(i-1)N/b}}{N/b} \qquad (1.3)$$

The mean of the b batches will give the estimate $\bar{Y}$ for the parameter(s) of interest.

$$\bar{Y} = \sum_{i=1}^{b} \frac{\bar{X}_i}{b} \qquad (1.4)$$

**REGENERATIVE PROCEDURES**

In the regenerative procedures "the idea is to identify random times at which the output stochastic process probabilistically "starts over", i.e., regenerates and to use these regeneration points to obtain independent random variables (r.v.'s) to which classical statistical analyses can be applied. This method was developed simultaneously by Crane and Iglehart (1974a, 1975a) and Fishman (1973, 1974), although the original idea of the regenerative method dates back to Cox and Smith (1961) and Kabak (1968).

Assume for the output process $\{Y_i, i \geq 1\}$ that there is a sequence of random indices $1 \leq B_1 < B_2 < \ldots$ called *regeneration points*, at which the process starts over probabilistically; i.e., the distribution of the process $\{Y_{B_j+i-1}$, $i = 1, 2, \ldots\}$ is the same for each $j = 1, 2 \ldots$, and the process from each $B_j$ on is assumed to be independent of the process prior to $B_j$. The portion of the process between two successive $B_j$'s is called a *regeneration cycle*, and it can be shown that successive cycles are independent and identically distributed (i.i.d.) replicas of each other. In particular, comparable r.v.'s defined over the successive cycles are i.i.d. Let $N_j = B_{j+1} - B_j$ for $j = 1, 2, \ldots$ and assume $E(N_j)<\infty$. If:

$$Z_j = \sum_{i=B_j}^{i=(B_{j+1}-1)} Y_i \tag{1.5}$$

the random vectors $U_j = (Z_j, N_j)$ are i.i.d. and provided that $E(|Z_j|) < \infty$, the steady state average response $\upsilon$ is given by $\upsilon = E(Z_j)/E(N)$...

We now briefly discuss how to obtain a point estimator for $\upsilon$ using the regenerative method. Suppose we simulate the process $\{Y_i, i \geq 1\}$ for exactly $n'$ regeneration cycles, resulting in the data $Z_1, Z_2, \ldots Z_{n'}$ and $N_1, N_2, \ldots N_{n'}$. Each of these sequences consists of i.i.d. r.v.'s; however, in general, $Z_j$ and $N_j$ are not independent. A point estimator for $\upsilon$ is the given by:

$$\hat{\upsilon}(n') = \frac{\bar{Z}(n')}{\bar{N}(n')} \tag{1.6}$$

Iglehart (1975), Meketon and Heidelberg (1982) and Glynn (1982) discuss alternative point estimators for $\upsilon$. " (Law, 1983).

## 1.2. SOME PROBLEMS IN SIMULATION

In this section we will describe some of the statistical problems found in the applications of both terminating and non-terminating simulations.

### 1.2.1. TERMINATING SIMULATIONS

The method of different and independent replications is used for the estimation of parameters in terminating simulations. However, the number of replications to be used is decided by the practitioner. Too few replications will not give an accurate estimate, and too many replications will be a waste of computer time. Although some procedures (discussed in Chapter 3) exist to determine the optimal number of replications, they are based on the "classical assumptions" of statistics that observations (or results from each replication) are identically distributed, independent, and follow a normal distribution. The two first assumptions are easily met, by the use of exactly the same parameters in all the replications, and by the use of different random number seeds for each replication. But the assumption of normality is not always met. Research is still being done on the effects of this lack of normality of the simulation results.

### 1.2.2. NON-TERMINATING SIMULATION

The analysis of non-terminating simulations, or "steady-state" simulations as it is sometimes called, is much more difficult than the analysis of terminating simulations. Among the problems that arise in this analysis two are worth mentioning:

#### 1. INITIALISATION BIAS PROBLEM.

A system is said to be in the steady state when the influence of the initial conditions has disappeared. Therefore, when the simulation is started from the initial state, these initial conditions will take some time to disappear. HOW LONG ? is one of the questions that the simulation practitioner has to answer. If steady state estimates are calculated while there is still some influence of the initial conditions, the estimate will be biased, and a confidence interval, if it is calculated, will be centred around the wrong value. A possible solution to this problem of the *Initialisation bias* is to delete a number N of the initial observations. However, the practitioner faces the new problem of finding a value for this number N. Several methods have been proposed in the literature. Studies of the state of the art by 1978 show that none of them is completely satisfactory (Wilson and Pritsker, 1978b). In the 1980's some new methods were proposed (Schruben,1982; Kelton and Law, 1983; Welch, 1983), but most of them are rather complex which makes them difficult to use by the user who is not familiar with advanced programming techniques (see discussion in Chapter 4). The usefulness of deleting some of the initial observations has been questioned. Fishman (1971) shows how this deletion will increase the mean square error and will greatly reduce the statistical reliability of the results but on the other hand Kelton and Law (1984), question Fishman's results and conclude that deletion of some of the initial observations is useful and effective. Further research is necessary using not only simple systems for which we can obtain an analytical answer but complex systems that are more representative of real world models in order to examine this question.

This problem appears mainly when the replications method is used for the estimation of steady state parameters. This is due to the fact that each replication starts with the same initial conditions which are usually not representative of the steady state conditions. When the estimation is made using a single but very long simulation run this problem of starting the simulation with the same initial conditions is almost eliminated but when the batch means method is used a new problem appears: that of the autocorrelation of the observations which is described in point 2 below.

## 2. AUTOCORRELATION IN THE OBSERVATIONS

This autocorrelation can be intuitively explained by an example. Suppose that the system being simulated is a queuing network. Obviously at a given time t, the time a customer has to wait is related to the past history of the system. If at time t-1 (assuming discrete event simulation) there are several customers waiting to be served, the probability of having to wait to be served for the new customer arriving at time **t** is much higher than if at time **t-1** there are no customers waiting to be served. "The effect of having autocorrelation among the data is to make it difficult to estimate the variation in the sample mean." (Seila, 1990) A direct consequence of this is that the estimated standard deviation calculated from the sample according to the classical formulas from statistics will usually be underestimated and consequently, any statistical test based on these values (mean and standard deviation) will be biased.

Due to this autocorrelation the main problem with the batch means method is to choose a batch size sufficiently large such that successive batches are independent. If they are correlated the variance estimator will be biased (either positively or negatively) and therefore the confidence interval (c.i.) thus calculated will be either too small or too large. This method is extensively discussed in Chapter 5.

Although the regenerative method does not present the problem of autocorrelation of the observations, it is not always possible, due to the characteristics of the method, to use it.

## 1.3. RESEARCH OBJECTIVES

Now that the main concepts and problems in the practice of simulation output analysis have been explained, we can discuss in more detail WHY research in this area is necessary, and what the main objectives of the research in this thesis have been.

Research in this area is important because as Alan Pritsker says:

"The analysis of simulation output is a perplexing topic. In practice, it appears that an analysis is either very easy or extremely difficult. Sometimes this dichotomy is hard to understand. Tremendous strides have been made in deriving theoretical results for output analysis and variance reduction. However, the results are not often used. The reasons for this are that the results are not easy to apply, thorough experimentation in the industrial and government sectors is not usually possible due to time constraints, the number of pitfalls associated with the applications of the results discourages their use and the number of variables and performance measures in a model make it difficult to apply the results. As an alternative, there has been a greater exploration of graphical means for viewing the outputs of a simulation...This has not solved the problem. What is needed is robust statistical techniques that can be applied to diverse systems." (Pritsker, 1989)

Having answered the question of WHY research in this area is necessary and useful, we can summarise the main objectives of the research reported in this thesis.

The first important aspect to take into account is that simulation may be used to solve some types of *real-world problems for which no analytical answer is available.* This means that the simulation "client" will not necessarily have special knowledge of the field. It would be different if simulation were going to be used only in the academic world (for example), where talk of "technical terms" and "complicated procedures" would be understood with no problems. We can suggest that the simulation user is "the person who interacts with the computer to enter a model and carry out simulations...Preferably the system user should be the same person who wants the results." (Symons, 1985).

Therefore, if simulation is going to be successful, it should be **"user-friendly"**. It means that "the user must be able to communicate this information" (the computer requires to give some information to produce the results) "quickly" (Symons, 1985). In other words, we must remember that the user is the person "who wants to use the system, not make the system, or follow long detours to reach this goal." (Symons, 1985). Unfortunately several of the procedures formulated to deal with problems in simulation are not easy to understand and to use.

A second aspect, also mentioned by Pritsker and which is important to take into account is the following: because simulation is used to obtain answers to "multifactor probabilistic problems", statistical analysis of simulation output is necessary in order to infer how accurate it is. This aspect of statistical analysis is quite important but unfortunately many times it is not taken into account. "Unfortunately many simulations are run without applying statistical analysis to the output...The simulation is run only once for each scenario to be analysed, a single value such as the average cost is computed and this number is treated as if it is the correct parameter value" (Seila, 1990). "In many simulation studies a great deal of time and money is spent on model development and "programming", but little effort is made to design appropriate simulation runs or to analyse correctly the resulting data." (Law, 1990). These are only two quotations among those found in the literature placing emphasis on the importance of a good statistical analysis of the simulation output data.

A third important point lies in the way research in simulation has been carried out. Frequently research is carried out in a specific area of simulation and new procedures are tested for very simple models for which an analytical answer exists. Usually the M/M/1 queue is used to test proposed procedures. (See for example Minh, (1989), and Kelton and Law, (1985) for just two examples of the use of the M/M/1 queue and similar to test new proposed procedures). However, the results thus obtained may be misleading because simulation should only be applied when systems are too complex to be analysed in a different way and where the elements of the system will interact with each other. This interaction does not occur when a simple system like the M/M/1 queue or any other simple system is used to verify a procedure. A typical statement that illustrates the use of only simple models for testing a proposed procedure is the following: "The primary purpose of the experiments is to assess the validity of the confidence interval estimates. Thus the models considered were restricted to those which could be solved by separable balance equation models" (Sauer, 1979). There are times when it has even been suggested that it is possible to obtain a "suitable approximation for the given stochastic model, and, second, we must calculate the asymptotic quantities of interest for the approximating model." (Whitt, 1989a).

In summary there are two main purposes for carrying out research in the area of simulation:

1. There is a need for simple and easily understandable procedures, not always available in those developed up to now.

2. The statistical analysis of simulation output data is many times overlooked but nevertheless, if simulation is going to give good results, this analysis should be included as part of the whole simulation process. (See Kelton (1983, 1985), Law (1980, 1982) for a discussion on statistical analysis of simulation output).

In this research we deal mainly with the first purpose, but using different simulation models for which an analytical answer cannot be calculated, we also discuss why an analysis previous to the simulation, as well as an analysis of the results from the simulation, is important. One of the fascinating aspects of research is that the answer to a question brings up several other questions, and while studying a phenomenon, several others come to light. In this way, when we were applying some statistical tools to the simulation output data obtained for terminating (or terminal) simulations, we found a simple, and easy to implement, answer to one of the puzzling questions in simulation. This point is discussed in Chapter 3.

In conclusion, in this thesis, we show how it is possible to design simple procedures to deal with some problems of simulation. Once these problems have been dealt with using procedures that work, but that are not time consuming, the time saved on collecting the data can be used on a thorough statistical analysis of it.

In order to test the general applicability of any procedure it is necessary to test it on several complex simulation models, rather than on simple models with none, or just very little interaction amongst the elements of the system. For this reason our research was applied mainly to simulation models for which no analytical answer can be calculated. Although the research described in this thesis is not theoretical from a mathematical point of view, it is more theoretical than empirical in terms of the models employed because these models do not correspond to real life simulations.

## 1.4. RESEARCH METHODS

In mathematical-related fields emphasis is usually placed in obtaining a mathematically accurate and exact solution. As pointed out by Newell, (1971): "Mathematicians working for their mutual entertainment will discard a problem either if they cannot solve it, or if being soluble it is yet trivial. An engineer concerned with the design of a facility cannot discard the problem...I have suffered many times the frustration of failing to solve elegantly what appeared to be a straightforward practical queueing problem, subsequently to discover that I could find very accurate approximations with a reasonable effort, and finally that I could obtain some crude estimates with almost no effort at all."

The same can be said about the way most problems in simulation have been tackled by simulation theoreticians. Procedures are proposed only if they have a mathematical background to support them. But one of the main characteristics of simulation is that it should be used ONLY when other approaches are not possible, and therefore no analytical solution can be found due to the complexity of the problem. Therefore, trying to formulate procedures having a strong mathematical background seems to be in contradiction with the very same nature of simulation.

Taking this into account, as well as the need to develop if possible "user-friendly" and simple simulation procedures, this research considers some of the problems that a practitioner is very likely to encounter in the application of simulation and shows that methods used to deal with these problems do not need to be complex and difficult to understand and to use.

However, to do this we are faced with a fundamental problem: we use simulation in those cases when an analytical answer cannot be found; for this same reason, it is very difficult to test the validity of a procedure that has been suggested to deal with a given problem. There are two possibilities in this case:

1. Test the procedure against several stochastic systems for which an analytical answer exists and if the procedure works reasonably well assume that it will work well in other complex systems.

2. Test the procedure in an empirical way against several simulation models (i.e., models for which no analytical answer can be calculated) and if

there are no contradictions, i.e., if all the systems perform according to the hypothesis that has been stated, it is reasonable to assume that the procedure will in most cases perform well.

Although the first approach has been widely used in simulation, it may be misleading. It is well known that a system like the M/M/1 queue (a favourite system used to test procedures in simulation papers) not only is not a "typical" case of a real-world problem, but (and this is valid for most cases where an analytical answer is obtained) in order to make the problem "mathematically manageable" several assumptions have been made. Therefore, the analytical answer will be more an approximation to the real value, than the real value itself. For this reason, to test the new procedures we have followed in this thesis the second approach described above. This means, that along with a procedure, a hypothesis is formulated on what we may expect with respect to the behaviour and type of results of the procedure. This is tested against several simulation models and most of the analysis in this thesis is based on results obtained for these models. If the hypothesis that we have formulated performs well for several different types of simulation models, it should perform well for other models also. The specific topics discussed in this thesis are described in section 1.5.

### 1.4.1. JOE'S THEOREM

There are two important characteristics of the research reported in this thesis: 1. The fact that it uses an empirical (in the sense that it uses models for which no analytical answer can be obtained) rather than a mathematical approach, and 2. One of its objectives is to show that simple solutions and methods exist for dealing with some common simulation problems.

However, there is a risk in this approach, a risk that has been mentioned by Grassman, who is the author of different papers in the area of operational research, and particularly, in queuing theory. In his article "Is the fact that the Emperor wears No Clothes Subject Worthy of Publication?" he discusses the problem of "bias against simple methods" (Grassman, 1986). To this end he tells us that "I have worked in industry for a few years, and I helped to implement some of the most successful operational research projects for the company for which I worked. During this time I came to realize how important it is to keep things simple. It is good engineering practice to start with the simplest approximation one can get away with and add new features only if and when they are needed." Continuing with this line of thought he formulates a theorem called **Joe's Theorem:**

**Joe's Theorem:**

> *Nothing is published in the area of queuing theory unless it is mathematically interesting. Nothing is applied in industry unless it is mathematically trivial. Since trivial results are not interesting, and since results that cannot be applied are not useful, nothing useful will ever be published in queuing theory.*"

And as Grassman says, it is not that we think that mathematicians are useless; it is the opposite, they have greatly contributed to the development of not only mathematics, operational research and simulation, but of many other areas. However, and this is one of the messages of this thesis "mathematical models are useful and necessary, but they can never capture all features of the system they represent. Consequently there is no guarantee that the optimum of the model is also best in real life. This fact is almost always ignored in the theoretical literature, but it is essential for any successful application of operational research" (and we should add, of simulation). "In order to be successful, one should always start with the real-life system never with its model." (Grassman, 1988)

This is one of the reasons for the other important objective of this thesis: procedures should not be tested for simple models with analytical solutions, as these analytical solutions include too many assumptions and simplifications. Even though the procedures proposed in this thesis have been tested in a more general way, they still may not reflect completely the behaviour of a real-life system, but this can only be analysed by the simulation practitioner.

## 1.5. THESIS OUTLINE

This thesis describes research into some topics related to some problems, mainly in the area of steady state simulation.

In the Introduction, Chapter 1, some basic concepts and types of simulation have been defined. Also some of the problems that may be found when using simulation are briefly discussed.

Chapter 2 expands some of the ideas considered in Chapter 1, and some of the considerations that the simulation practitioner should make before the simulation is run.

Chapter 3 studies the problem of the number of replications that are required for the estimation of parameters in terminating and steady state simulations.

Chapter 4 gives a more detailed analysis of the Initialisation Bias Problem and proposes a method for dealing with it. This method deals with this problem by eliminating some of the initial observations which are not representative of the steady state conditions.

Chapter 5 discusses the Batch Means Method for the estimation of steady state parameters and shows how some of the procedures proposed in the literature up to now for the estimation of steady state parameters using this method do not work well in practice.

In Chapter 6 we propose another method to deal with the initialisation bias problem. Instead of deleting any of the initial observations, the method proposed in this chapter is based on the assignment of different weights to the observations recorded from the simulation output: initial observations are assigned smaller weights.

Finally, Chapter 7 presents the conclusions and areas for future research.

## 1.6. SUMMARY

This chapter has defined what simulation is, when and where it could be used and which are the practical problems that a simulation practitioner is likely to face. Two main objectives for this research have been identified: the need for simple and easy to understand procedures and the need for testing any proposed procedure with complex, and not only with simple, simulation models. Another important point to be considered is that these procedures should be statistically robust and this can only be confirmed when they have been applied to different simulation models. In this thesis we try to develop approaches that might successfully contribute to these points. In case that there are still some doubts on how useful the simulation approach is to solve problems let me quote Pritsker (1989):

""We have commercialised the field and demonstrated, without a doubt, the benefits obtainable from modelling, analysis and problem solving using simulation.

In 1947 Winston Churchill in a speech before the House of Commons presented the following view of democracy:

"Many forms of government have been tried, and will be tried in this world of sin and foe. No one pretends that democracy is perfect or all-wise. Indeed, it is the worst form of Government, except all those other forms that have been tried from time to time. " (Churchill, 1947)

I close by paraphrasing Churchill's statement:

*No one pretends that simulation is perfect. Indeed, it has been said that simulation is the worst form of analysis except all those other forms that have been tried from time to time.""*

# CHAPTER 2 : RESEARCH DISCUSSION

## 2.1. INTRODUCTION

The main topics, and purposes of the research described in this thesis are discussed in Chapter 1. We expand some of these ideas in this chapter in order to create the scenario required for the discussion that follows in the remaining chapters. We also discuss in this chapter some important aspects that should be considered by the simulation user before running the simulation, and some points that should be taken into account in the analysis of the simulation output.

### 2.1.1. CHAPTER OBJECTIVES

While in Chapters 3 to 6 we deal with specific problems of simulation and formulate solutions that are easy and simple to implement, in this chapter we try to give a general view of simulation and some practical aspects about its application.

One of the main characteristics of the research presented in this thesis is the use of complex simulation models for which no analytical answer can be calculated. Such an approach requires the appropriate scene setting for it to be understood. We intend in this chapter to create such a scenario by considering some practical aspects of the use of simulation. This practical consideration is important because the good practice of simulation requires some previous analysis of the system to be simulated.

Simulation is much more than just running a program on a computer and recording the results. If simulation is going to give accurate and acceptable solutions for a given problem, it is important to analyse the system to be simulated before running the simulation. This analysis prior to the simulation is sometimes omitted but, as shown in this chapter it may save valuable time; for example, if the objective of the simulation is to estimate the steady state parameters, the simulation practitioner should consider before running the simulation if such a state can exist.

These types of practical consideration concerning the use of simulation are given in this chapter. They are a complement of the approach used in this thesis, and create the appropriate framework for the treatment of simulation in the following chapters.

### 2.1.2. CHAPTER OUTLINE

A first point to consider (Section 2.2.) is the nature of the research, as this is empirical more than theoretical. As discussed in Chapter 1 simulation is used only when an analytical solution cannot be found; for this reason we do not propose procedures supported by a sound mathematical theory, but by a good empirical performance over a variety of simulation models.

Because of the characteristics of the simulation software (VS6) used (Paul and Chew (1987); Crookes et al, 1986), most of the research reported in this thesis has been conducted in the area of **queuing networks**. Section 2.3. discusses some of the characteristics of these networks.

In steady state simulations, a simple analytical analysis of the system helps the practitioner to determine if such a steady state exists or not. This point is discussed in Section 2.4.

One of the problems discussed in this thesis is that of the influence of the initial conditions, or *Initialisation Bias Problem*. Two procedures (See Chapters 4 and 6) are proposed to deal with this problem but Section 2.5. presents a general discussion about the cases when such a problem may not exist or may not be eliminated because of the characteristics of the system being simulated.

Section 2.6. discusses some problems found in the statistical analysis of the simulation output like, for example, that of a large standard deviation as compared to the sample mean.

## 2.2. WHY SHOULD WE USE SIMULATION MODELS FOR WHICH NO ANALYTICAL ANSWER CAN BE CALCULATED ?

The first characteristic of the research described in this thesis is that it is not theoretical from a mathematical point of view, but it is based on results obtained for some complex simulation models. Based on practical observations or on a theoretical analysis of the behaviour of simulation models, we formulate a *hypothesis or propose a new procedure*, which usually has no rigorous mathematical support.

In order to study how well this hypothesis works, and how useful it is for the solution of the problem under consideration we use different complex simulation models and, as a further check, some commonly used systems with known analytical answer. We expect that the new proposed procedure will provide a "good" solution to the problem of interest in all the different models. Sometimes, when we expect the procedure to have some limitations in its application, we may also assess it with results obtained from the simulation. What "good" means depends on the particular problem under consideration.

Although this non-mathematical approach may not be approved of by all simulation theoreticians, it is useful because it permits the identification of simple and easy to understand methods. The use of results obtained from the simulation, as shown in this thesis, will highlight simple facts of otherwise complex problems.

A second advantage of using results obtained for different types of simulation models is that sometimes common facts to different models can be identified and procedures that do not require the setting of values of certain parameters that may be model dependent can be proposed. This point is discussed in detail in Chapters 4 and 5.

A third advantage of the approach used in this thesis is that carrying it out shows that the necessary modifications to the simulation software can be done. Another advantage is that a procedure that, even if giving "good" results, may be of little practical use because of the relatively large computer time it requires, may be easily identified.

## 2.3. QUEUING NETWORKS

In this section we describe the concept of **queuing networks**, and some parameters of interest (queuing time and queue length) in these networks. The first step to follow when simulation is used, is to obtain a *model* of the real-world system to be simulated. There are different approaches concerning the type of model to use and its choice depends on the simulation practitioner and on the software available. A possible way of describing a system is by defining the *entities,* or elements, of the system; if necessary, individual elements belonging to a given class of entity can be identified by assigning an *attribute* to them. The entities are either **engaged** in an *activity* or are **"idle"** waiting to start an *activity.* The type of models that use *activities* and *entities* in their definition, are especially suited to the description called *queuing networks* (See Section 2.3.1.).

The use of queuing networks is not limited to the area of simulation. They also have been used in theoretical studies to model the contention for resources, which is usually the dominant factor in the performance of computing and communications systems. Some examples of studies in this field can be found in Ayani, (1989), Ayani and Rajaei, (1990), Chandy and Sherman, (1989), De Vries, (1990), Lin and Lasowska, (1989), Reed et al, (1988), Wagner and Lasowska, (1989). References to studies previous to 1980 can be found in the bibliography. However, most of these references study queuing networks as a Markovian process and under some assumptions try to find approximate solutions. Some efforts have been put into the solution of queuing networks by simulation. Examples of these studies can be found in Rypley, (1988), Glynn, (1988), and Schruben and Yucesan, (1988). Other references of studies are given in the bibliography. Studies to assess the validity of the confidence interval estimates for queuing models of computer systems have also been carried out by Amer (1982), Mamrak (1980), and others.

### 2.3.1. STATES OF A QUEUING NETWORK. DEFINITION

The *State* of a system is the set of variables needed to describe the system at any time.

In the simulation of a post office, for example, the possible states are the number of customers waiting to be served at a given time and the number of busy clerks. Similarly, in a production system where the machines that perform the work can break down, the possible states of the system are their status: busy, idle, or down. Notice that the word "possible" is underlined to suggest that the variables required to describe the behaviour of the system will ultimately depend on the objectives of the study.

There are two basic types of states in a queuing network:

a. An *Active State*, also called an **activity**, which requires the co-operation of different classes of entity. One characteristic of an *active state* is that its duration is known beforehand. This duration can be either deterministic, or can be sampled from a specific probability distribution. One example of an *active state* is, in the simulation of a post office, the period of time while a customer is serviced at the counter by a post office clerk.

b. A *Dead State*, also called a **queue** is a state in which the entity waits for an activity to start. Its main characteristic is that it does not require the co-operation of different classes of entity. In contrast with an *active state*, the length of time that an entity remains in a *dead state* cannot be known beforehand. It will depend in general on the interactions of the different entities in the system.

Appendix A explains a way of modelling queuing networks in simulation using what is called **Activity Cycle Diagrams** (A.C.D.). These diagrams show for each entity a cycle of active and dead states that for the sake of clarity will alternate, as is explained in Appendix A. Most of the study reported in this thesis will refer to queues used in the A.C.D.'s of some systems for which no analytical answer exists.

## 2.3.2. SOME PARAMETERS OF INTEREST IN A QUEUING NETWORK

In a queuing network the entities or elements of the system are either engaged in an **activity** or **waiting** to start it. Therefore, the model used to represent the system should describe these queues and these activities. Some of the steady state parameters that may be of interest in this type of model are the **queuing time** and the **queue length**.

**DEFINITION**

a. **Queuing time** is the average time that a unit of an entity type or class waits in a queue to start an activity.

b. **Queue length** is the average number of units belonging to a given class of entity that are waiting in a queue.

Therefore the **steady state mean queue length** is the number of units likely to be found at any time in the queue, provided that this time is very large as compared to the time when the system started its operation. Similarly, the **steady state mean queuing time** is the average time that any entity will spend in a queue waiting for an activity to start. In unstable systems these two parameters will never reach a steady state, but their value will increase with an increase in the simulated time.

These parameters are important in a queuing network because they can be used to make inferences about the behaviour of the system. The procedures proposed in this thesis are verified using results for these two parameters, but they can be easily modified and extended to other types of simulation models.

## 2.4. HOW TO IDENTIFY CRITICAL QUEUES

As discussed in Section 2.2. one of the important aspects of the research described in this thesis is its empirical nature, in the sense that we are not developing mathematical supported procedures but using results obtained from simulation to infer something concerning the problem of interest. We are using such an approach to show that the practice of simulation does not require the use of complex procedures, but that simple methods can also provide a satisfactory answer. However, one of the important aspects of simulation, that is many times not taken into account, is that of the need of a previous analysis of the system to be simulated; usually this analysis can show some characteristics that make the use of simulation unnecessary. If the interest of the simulation is to estimate some steady state parameters, we should first check, in a quick way, if such a state exists. If such a state exists, the practitioner can identify in some cases those queues that may take longer than others to reach the steady state. How to carry out such an analysis is discussed in this section.

The case where the parameters of a queue may never appear to reach a steady state is discussed in Section 2.4.1. In this sub-section, the systems that may present this "odd" behaviour (no steady state) are identified. We also give some guidelines on how to identify possible queues with this problem.

In other cases, depending on the values of the simulation input parameters, like for example the time that the different activities take to be executed (called in this thesis "execution time"), the number of "servers" in each queue, etc, some queues may take a long time to reach a steady state; section 2.4.2. gives some practical guidelines to help the practitioner in the identification of some queues that **may** require a long simulation run length to reach the steady state.

### 2.4.1. QUEUES THAT MAY NEVER APPEAR TO REACH A STEADY STATE.

Systems can be classified as follows according to the relation of their entities to the "outside" world:

1. "<u>CLOSED</u>" systems, for example the STEELWORKS (see Appendix A). In this type of system the number of units of each one of the entities is limited and is defined at the beginning of the simulation. In other words, they are systems that consist only of *permanent* entities that are always part of the

system. In this case all the queues will eventually reach the steady state because these systems have limited "resources" and therefore the different parameters of the queues will never grow without bound. The system is self-balancing or self-regulating.

2. "<u>OPEN</u>" systems, are systems where some of the entities are *permanent* and others are *temporary*. These *temporary* entities come from the "outside world" and once they have completed their life cycle they go back to the "outside world" becoming an element of no further interest. In this case, when the approximate value of the *traffic intensity* $\tau = \lambda/s\mu$ (s is the number of "servers" serving at a rate $\mu$ each) for one of the queues belonging to entity A is greater or equal to 1, there is no steady state for one or more of the queues belonging to that particular entity (A). It should be noted that queues belonging to permanent entities cannot have their queue length (and therefore their queuing time) increased without bound; therefore, they will eventually reach a state of equilibrium, albeit this is 100% utilisation.

**NOTE** Although the traffic intensity is usually referred to by the greek letter $\rho$, in this thesis we have used the greek letter $\tau$ to refer to it.

It is important to notice that in practice there is no difference in simulation between "closed" and "open" systems, as an "open" system is modelled as a close one for the sake of simplicity. However, the difference is important in our discussion: in an "open" system, the queue length can (at least theoretically) increase without bound. This is not possible in a "closed" system. But at the same time, "closed" systems will be stable if the parameters are time invariant. An example of non time invariant system occurs when a server is ageing and cannot always serve at the same rate.

The queues of temporary entities with an infinite supply of units may be unstable and the relation $\lambda/s\mu$ should be determined in order to check for instability. Examples of this behaviour are found in the LAUNDERETTE, the FISH PACKING SYSTEM, the BRAZILIAN HOSPITAL and the PUB, among others. These simulation models, which are used throughout this thesis, are described in Appendix A. Two main reasons for the relation $\lambda/s\mu$ to be greater or equal to 1 are discussed in **hypothesis 1** and **hypothesis 2**.

HYPOTHESIS 1

*A relatively long time of execution of one of the activities, or a small interarrival time (of temporary entities) may cause instability of one or more of the queues belonging to temporary entities.*

Clearly when an activity has a relatively long execution time the value of $\mu$ is smaller and the relation $\lambda/s\mu$ gets larger. Similarly, a small interarrival time implies a large value of $\lambda$ and an increase in the relation $\lambda/s\mu$. To illustrate this point we use a simulation model of a LAUNDERETTE system.

1. THE LAUNDERETTE.

This system is described in Section A.2.2., Appendix A. The corresponding A.C.D. is given in Figure A.5. In order to explain how a long activity execution time, as compared with that of other activities may cause instability of one queue, the LAUNDERETTE system is simulated for the different conditions shown in Table 2.1. This table gives the execution time for the different activities of the launderette model. For example, the arrival of customers to the system is assumed to follow a negative exponential distribution (NEGEXP) with mean interarrival time of 8 minutes.

Similarly, the execution time of the TRANSPORT activity is sampled from a uniform distribution (UNIF) that takes values between 1 and 5. From this table we see that the execution time of the activity LOADD increases to 20 minutes in condition 2.

The number of units of the different permanent entities is the same for both conditions, 1 and 2:

| ENTITY | Number of Units |
|---|---|
| Washing machines | 7 |
| Baskets | 8 |
| Driers | 2 |

| Activity | Execution time Condition 1 | Execution time Condition2 |
|---|---|---|
| ARRIVAL | NEGEXP(8) | NEGEXP(8) |
| LOADW | 40 | 40 |
| UNLOADW | UNIF(3,5) | UNIF(3,5) |
| TRANSPORT | UNIF(1,5) | UNIF(1,5) |
| LOADD | 4 | 20 |
| DRY | NORMAL(10,4) | NORMAL(10,4) |

**Table 2.1.** Probability distributions for the execution time of the different activities of the LAUNDERETTE model, for two different conditions.

Under these conditions the arrival rate of "customers" to the LAUNDERETTE is $\lambda = 7.5$ per hour and if we look at the service rate for the LOADD activity, $\mu_1 = 15$ per hour (condition 1) and $\mu_2 = 3$ per hour (condition 2). The number of servers (driers in this case) is s = 2. In the second case $\lambda/s\mu_2 > 1$ and therefore we can expect one or more of the queues belonging to the entity customer to become unstable. *Without any need for simulation, but with a simple analysis of the A.C.D,* the simulation practitioner can easily conclude that the only possible unstable queue is the WASHQ because the maximum queue length of the other queues belonging to the entity customer is limited in number by the number of baskets.

This analysis can be confirmed if the system is simulated for the two different conditions of Table 2.1. Table 2.2. shows the results corresponding to the mean queuing time of the DRYQ and the WASHQ queues as function of the simulation run length. From these results it is apparent that when the LOADD activity execution time is 20 the WASHQ parameters do not reach a steady state.

While numerical data is important it is not always easy to study and to draw conclusions from it, especially if we are comparing two different models where the numerical range is different. For this reason sometimes we use a graphical approach based on the data shown in tables for the analysis of the examples discussed in this thesis. The data is presented in a LINE graph of the mean estimates (Y-axis) as a function of the simulation run length (X-axis). As our objective in this example is the analysis of a simulation model under two different conditions and the numerical information obtained lies in different ranges, we have used two Y graphical scales for comparing those results. Some graphical software, like HARVARD 3.0 or QUATTRO (Release 4.0) (used in the graphs of this chapter) has this option. In other words, we use two Y axes when we want "to compare series that use different units of measure or that vary greatly in magnitude." (Harvard Graphics 3.0 User's Manual, 1990).

For this example of the LAUNDERETTE, the graphs drawn for the mean queuing time of the WASHQ and the DRYQ queues as a function of the simulation run length are given in Figures 2.1. and 2.2. In these graphs one of the Y-axes corresponds to a LOADD activity execution time of 4 and the other to an execution time of 20. From these graphs it is clear that the mean queuing time and, therefore, the mean queue length, for the WASHQ queue converges to a steady state value as the simulation run length increases when the LOADD activity takes on average 4 minutes to be executed.

On the other hand, when this activity takes 20 minutes to be executed there is an increase in the mean value corresponding to an increase in the simulation run length which means that there is no convergence to a steady state value. For the DRYQ, no matter how long the activity LOADD takes to be executed (4 or 20) there is always convergence to a steady state.

The values for the standard deviation of these estimates are not shown in these tables but it is interesting to notice that when a queue parameter does not reach a steady state, the standard deviation of the estimate increases when the run length increases. On the other hand, when a steady state exists for a parameter of a queue, the standard deviation will tend to decrease as we increase the run length. This may not be always true for short run lengths, when there is still some influence of the initial conditions. However, for large run lengths there is a reduction in the standard deviation of the estimate.

| | WASHQ Mean Queuing Time Estimates | | DRYQ Mean Queuing Time Estimates | |
|---|---|---|---|---|
| Run Length | LOADD: 4 | LOADD: 20 | LOADD: 4 | LOADD:20 |
| 1500 | 5.644 | 246.279 | 12.628 | 85.064 |
| 4500 | 6.787 | 932.891 | 14.601 | 90.542 |
| 7500 | 7.352 | 1625.226 | 15.529 | 91.483 |
| 10500 | 7.220 | 2320.230 | 15.362 | 91.955 |
| 13500 | 7.336 | 3022.872 | 15.494 | 92.165 |
| 16500 | 7.287 | 3721.846 | 15.477 | 92.328 |
| 19500 | 7.180 | 4418.866 | 15.409 | 92.432 |
| 22500 | 7.340 | 5117.860 | 15.501 | 92.512 |
| 25500 | 7.449 | 5819.596 | 15.619 | 92.546 |
| 28500 | 7.467 | 6518.196 | 15.636 | 92.591 |

**Table 2.2. WASHQ and DRYQ** Mean Queuing Time Estimates as a function of the simulation run length and of the **LOADD** activity execution time.

Similar examples where instability of the system is attributable to a value of $\lambda/s\mu>1$ for different simulation models are given in Appendix B. Another possible reason for a value of $\tau = \lambda/(s\mu)$ to be larger than 1 in a simulation is the following:

<u>HYPOTHESIS 2</u>

*When the number of servers (i.e., barmaids in the PUB model) is small, some of the queues belonging to temporary entities that use that server may become unstable.*

When the number of servers is small the value of $\lambda/s\mu$ may be large and this may lead to instability of one or more of the queues, belonging to the *temporary* entity, and that are served by this particular type of server. Numerical examples illustrating this case are included in Appendix B.

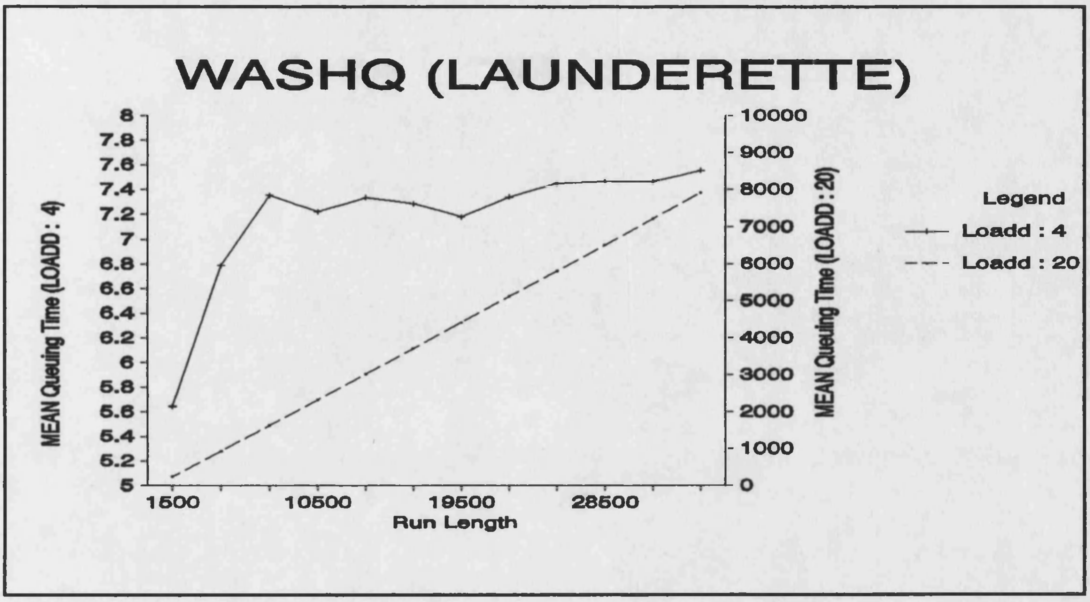



**Figure 2.1.** WASHQ mean queuing time estimates as a function of the simulation run length and of the LOADD activity execution time.

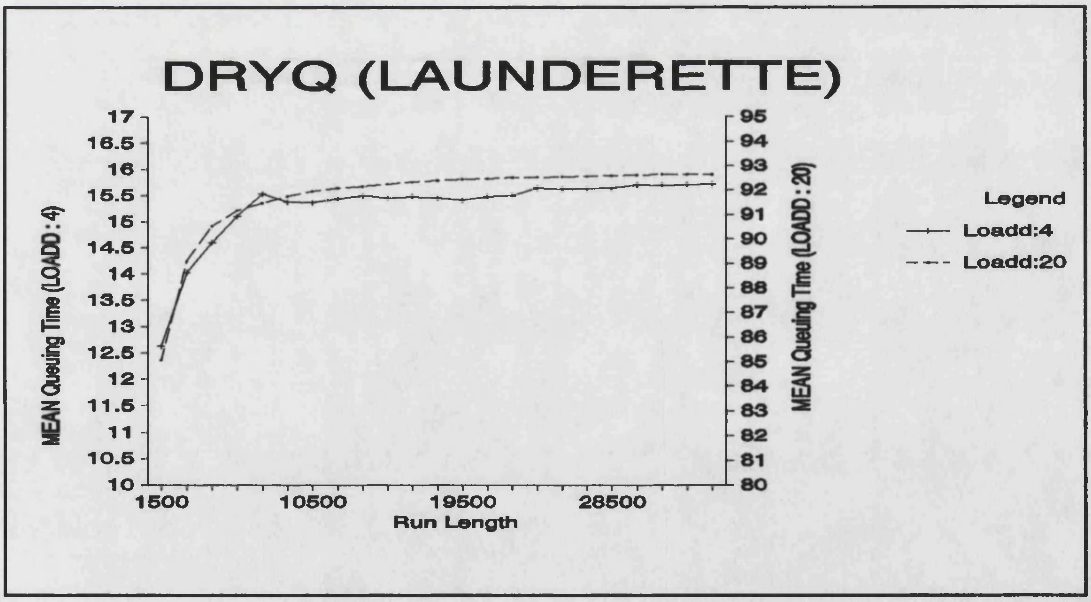



**Figure 2.2.** DRYQ mean queuing time estimates as a function of the simulation run length and of the LOADD activity execution time.

## CONCLUSION

HYPOTHESES 1 and 2 suggest that a previous study of the model can and should be done by the simulation user or practitioner before spending computer time trying to find a steady state that does not exist. In this way, if one of the queues of the simulation model is unstable, an analysis previous to running the simulation program can detect the problem and on the other hand, if such an analysis is not carried out, the study of the "wrong" queues (for example DRYQ) may lead to erroneous conclusions that a steady state exists for the system under consideration.

### 2.4.2. QUEUES THAT MAY TAKE A RELATIVELY LONG TIME TO REACH A STEADY STATE

Closed systems like for example the STEELWORKS, can be called "well-behaved" in the sense that they will always reach a steady state and with no extreme values of activity execution times or number of units of one of the entities of the system, most queues will reach the steady state for not too long simulation run lengths. However, in open systems, some queues may take considerably longer to reach the steady state depending on the entity they belong to, the type of input distribution, etc. In this sub-section we will try to give some guidelines for the identification of possible critical queues, where by "critical" we mean those queues that may require a long simulation run length for the parameters, for example the mean queuing time and the mean queue length, to reach the steady state. The study will be divided in two parts:

1. Possible critical queues for temporary entities.
2. Possible critical queues for permanent entities.

We will illustrate the conclusions of this sub-section with an example corresponding to the PUB simulation model; Appendix B includes additional results for the LAUNDERETTE and the FISH PACKING simulation models.

#### 1. POSSIBLE CRITICAL QUEUES FOR TEMPORARY ENTITIES

In this type of entity, the units "arrive" to the system from the "outside world" and once they have finished their life cycle they are of no interest any more. From queuing theory, in most cases, random arrivals are modelled by a negative exponential distribution. This is a highly skewed distribution; as reported by Andrews et al (1972), who studied the robustness of about 70 different point estimators of location, the sample mean is very sensitive to outliers. These are values much larger or much smaller than the rest of the values in the data set. Skewed distributions are more likely to produce outliers in the simulation output data. Therefore, we can expect that those queues used in the A.C.D. to model the arrival of a temporary entity to the system, will require long simulation run lengths to reach the steady state, especially if the arrival of the entities is modelled using a negative exponential, or any other highly skewed distribution.

How critical are other queues belonging to temporary entities is a question that does not have an easy answer. In general, due to the interaction of the different entities, how long is the simulation run length required for these queues to reach the steady state will also depend on the number of units of the permanent entities with which the temporary entity interacts, and on the time that the activity (or activities), in which that particular queue is involved, takes to be executed.

## 2. POSSIBLE CRITICAL QUEUES FOR PERMANENT ENTITIES

In the case of permanent entities, an important factor on how long is the simulation run length required for the parameters of the queue to reach the steady state is the number of units of the entity. This is true especially for those entities that are described in the A.C.D. by a single queue, like for example the entity **barmaid** in the PUB model, which is described by a single queue, called IDLE in the A.C.D. (See Figure A.4) In this case, we can expect that the larger the number of **barmaids** in the system, the shorter the simulation run length required for the queue of that particular entity to reach the steady state. The main reason for this behaviour is that as the number of "servers", s, (i.e. number of units of a permanent entity) increases, the value of the traffic intensity, $\tau=\lambda/(\mu*s)$ decreases. As the traffic intensity decreases we can expect a better "behaviour" and therefore, with a shorter simulation run length the parameters of the queue will reach the steady state. On the other hand, as $\tau$ increases and approaches 1, the simulation run length required for the queue to reach the steady state becomes longer. With respect to other permanent entities that are represented by more than one queue in the A.C.D. how long is the simulation run length required for the parameters of the different queues to reach the steady state needs some analysis of the characteristics of the system, as well as of the A.C.D.

In Section 2.4.3. we give results corresponding to some of the queues of the PUB simulation model, which has been simulated under different conditions in order to illustrate the points discussed in this sub-section.

### 2.4.3. SIMULATION OF THE PUB MODEL UNDER DIFFERENT CONDITIONS

The PUB model has been simulated for three different conditions in order to study the influence of the number of units of permanent entities and of the distribution used for modelling the arrival of customers to the system. The following conditions have been used in the different experiments:

| Condition | Arrival | Number of BARMAIDS |
|---|---|---|
| 1 | Negative exponential, mean 15 | 3 |
| 2 | Normal, mean 12 and variance 16 | 3 |
| 3 | Negative exponential, mean 15 | 8 |

For each one of the three conditions the mean queuing time estimates of the WAIT (entity : customer), IDLE (entity : barmaid), and CLEAN (entity: glass) queues as a function of the simulation run length have been obtained. The number of glasses in the system is 50 in all the three different conditions.

Our objective in these examples is to show that, under some conditions, some of the queues will require a longer simulation run length to reach the steady state. But however, to show this we face the problem that in most of the simulation models that have been used throughout this thesis (like for example the PUB) the steady state is unknown. Nevertheless, the real value can be obtained if the model is simulated for a very long simulation run length and using a very large number of replications. This approach would not be feasible in real life due to the extremely large computer time that it requires. As discussed in more detail in Section 2.6.2. we will assume that the parameter has reached the steady state when the mean estimates fall within 2.5% of the real value that we have obtained by simulating the model for a very long simulation run length. We have chosen 2.5% as we consider that talking of a queue length of 2.0 or a queue length of 2.05 (2.5% of increase) provides, from a practical point of view, the same information. In the rest of this sub-section we will present and analyse the results obtained for the different conditions under which the PUB model has been simulated.

CONDITION 1

In this case we have simulated the arrival of customers to the system using a negative exponential distribution. Therefore, we can expect, according to the discussion of Section 2.4.2. that the WAIT queue will require a long simulation run length to reach the steady state.

Similarly, because the number of **barmaids** is not too large as compared to the number of units of the other permanent entity in the system, the **glasses**, we can expect that a rather long simulation run length is required for the parameters of the IDLE queue to reach the steady state.

Finally, as the number of **glasses** in the system is rather large (50) we can expect that the CLEAN queue will reach the steady state for very short simulation run lengths.

In Appendix C we obtained the following steady state values for the mean queuing time of the queues of interest in this model:

| Queue | Steady state value |
|---|---|
| WAIT | 1.141 |
| CLEAN | 209.400 |
| IDLE | 2.001 |

Table 2.3. gives the mean queuing time estimates for the three queues as a function of the simulation run length. We have underlined the mean estimates for which the parameters can be considered to have reached the steady state. This means that for the simulation run lengths corresponding to the underlined values, as well as for longer run lengths, the mean estimates all fall within 2.5% of the real steady state value.

As we expect, the WAIT mean queuing time takes a long simulation run length, in fact longer than 25000 minutes, for the mean estimates to fall within 2.5% of the steady state value.

The IDLE mean queuing time, although it requires a long simulation run length to reach the steady state, approximately 20000, gets stable sooner than the WAIT mean queuing time parameter.

The CLEAN mean queuing time estimates fall within 2.5% of the steady state value for a very short simulation run length (2000), as was also expected.

CONDITION 2.

In this case the arrival of customers to the system has been modelled using a normal distribution with mean 12 and standard deviation 4. Table 2.4. gives the mean queuing time estimates for the three different queues of interest in this model, as a function of the simulation run length, based on 100 replications. The steady state values, obtained by simulating the system for a very long period of time, are the following:

| Queue | Steady state value |
|---|---|
| WAIT | 1.165 |
| CLEAN | 155.700 |
| IDLE | 0.502 |

| PUB - Critical queues. | | | |
|---|---|---|---|
| | Mean queuing time estimates | | |
| Run Length | WAIT | CLEAN | IDLE |
| 1000 | 0.999 | 195.319 | 2.358 |
| 2000 | 1.007 | 205.204 | 2.206 |
| 3000 | 0.982 | 208.675 | 2.176 |
| 4000 | 1.020 | 208.962 | 2.126 |
| 5000 | 1.005 | 211.577 | 2.152 |
| 6000 | 1.023 | 212.423 | 2.164 |
| 7000 | 1.027 | 212.814 | 2.150 |
| 8000 | 1.043 | 212.280 | 2.133 |
| 9000 | 1.035 | 212.397 | 2.124 |
| 10000 | 1.034 | 212.322 | 2.103 |
| 11000 | 1.047 | 211.898 | 2.086 |
| 12000 | 1.053 | 211.072 | 2.067 |
| 13000 | 1.053 | 211.059 | 2.073 |
| 14000 | 1.063 | 210.800 | 2.059 |
| 15000 | 1.059 | 210.901 | 2.055 |
| 16000 | 1.054 | 211.316 | 2.064 |
| 17000 | 1.055 | 211.276 | 2.057 |
| 18000 | 1.054 | 211.314 | 2.055 |
| 19000 | 1.057 | 211.320 | 2.055 |
| 20000 | 1.063 | 211.047 | 2.048 |

**Table 2.3.** Mean queuing time estimates for the WAIT, the IDLE, and the CLEAN queues, when the arrival is modelled with a negative exponential distribution with mean 15, and there are 3 **barmaids** in the system.

We have underlined in Table 2.4. those values for which the mean estimates start falling within 2.5% of the steady state value. The results of this table confirm what we expected: when the arrival distribution is rather symmetric, the simulation run length required for the WAIT queue parameters to reach the steady state is not as long as it would be in the case of a skewed distribution, like the one used in condition 1 above. In this case, the mean queuing time of the WAIT queue requires a simulation run length of 9500 minutes to reach the steady state. When the arrival was modelled using a negative exponential distribution a simulation run length longer than 25000 required for the parameters of this queue to reach the steady state.

Similarly, with a large number of glasses in the system we can expect the CLEAN mean queuing time to reach the steady state for a short simulation run length (1000 in this case).

With a mean interarrival time of 12 minutes, we can expect the barmaids to be rather busy, and therefore, as the initial state is **idle** for the barmaids, there will be a rather large change in this state. This implies that the simulation run length required for the mean queuing time of the IDLE queue to reach the steady state will be rather long as the larger the change in the initial state of the queue, the longer the time it takes to get stable. Or using queuing theory, the smaller the value of the number of barmaids ($s$) the larger the value of the traffic intensity, $\lambda/(\mu s)$. In this case we notice, from the underlined values in Table 2.4. that the IDLE mean queuing time requires a simulation run length of at least 18000 minutes to reach the steady state.

## CONDITION 3.

In this condition the arrival is modelled using a negative exponential distribution with mean 15 but we have increased the number of barmaids in the system from 3 to 8. Therefore, we expect the mean queuing time of the WAIT queue to be very small, as very few customers will have to wait to be served. This means that the change in the initial state of the queue, which is **empty**, will be small, and the steady state should be reached for a very short simulation run length. Similarly, due to the rather large number of barmaids in the system, the change in the initial state of the IDLE queue will not be large (or using queuing theory, the traffic intensity will be small), and again the steady state should be reached for a short simulation run length. We obtained the following approximate values for the real mean steady state values:

| Queue | Steady state value |
|---|---|
| WAIT | 0.005 |
| CLEAN | 232.000 |
| IDLE | 14.510 |

Table 2.5. gives the mean queuing time estimates as a function of the simulation run length. In this table the underlined values correspond to the minimum simulation run length required for the mean estimates to fall within 2.5% of the real steady state value. Just as expected, the mean queuing time estimates of the three different queues require very short simulation run lengths to reach the steady state.

## 2.5. INITIALISATION BIAS PROBLEM

One of the problems in the estimation of steady state parameters, when the replications method is used, is that of the simulation run length. If it is not long enough there will be some influence of the initial conditions still present and the estimate will be biased. This problem, known as the *Initialisation Bias Problem*, has been discussed in Chapter 1, and methods to deal with it are discussed in Chapters 4 and 6. However, before using one of these methods, or any other of the methods that have been proposed for the elimination of this problem, the practitioner should consider if it is necessary or possible to eliminate it.

| PUB - Critical queues, ARRIVAL : NORMAL(12,4) | | | |
|---|---|---|---|
| | Mean queuing time estimates | | |
| Run Length | WAIT | CLEAN | IDLE |
| 1000 | 1.004 | 155.067 | 0.767 |
| 2000 | 1.022 | 158.433 | 0.666 |
| 3000 | 1.057 | 158.509 | 0.623 |
| 4000 | 1.079 | 158.412 | 0.603 |
| 5000 | 1.115 | 157.674 | 0.588 |
| 6000 | 1.113 | 157.360 | 0.567 |
| 7000 | 1.110 | 157.109 | 0.558 |
| 8000 | 1.111 | 156.988 | 0.546 |
| 9000 | 1.129 | 156.143 | 0.532 |
| 9500 | 1.138 | 155.874 | 0.530 |
| 10000 | 1.145 | 156.008 | 0.531 |
| 11000 | 1.150 | 156.197 | 0.530 |
| 12000 | 1.142 | 156.303 | 0.527 |
| 13000 | 1.137 | 156.512 | 0.526 |
| 14000 | 1.141 | 156.496 | 0.525 |
| 15000 | 1.141 | 156.398 | 0.522 |
| 16000 | 1.139 | 156.188 | 0.521 |
| 17000 | 1.136 | 156.020 | 0.518 |
| 18000 | 1.137 | 155.572 | 0.513 |
| 19000 | 1.134 | 155.595 | 0.514 |

**Table 2.4.** Mean queuing time estimates for the WAIT, the IDLE, and the CLEAN queues, when the arrival is modelled with a normal distribution with mean 12, and there are 3 barmaids in the system.

| PUB - Critical queues, ARRIVAL : NEGEXP(15) ; 8 BARMAIDS. | | | |
|---|---|---|---|
| | Mean queuing time estimates | | |
| Run Length | WAIT | CLEAN | IDLE |
| 500 | 0.003 | 186.946 | 15.824 |
| 1000 | 0.004 | 209.546 | 14.993 |
| 1500 | 0.004 | 217.989 | 14.944 |
| 2000 | 0.004 | 221.890 | 14.828 |
| 2500 | 0.004 | 224.026 | 14.791 |
| 3000 | 0.005 | 226.460 | 14.842 |
| 3500 | 0.005 | 226.826 | 14.739 |
| 4000 | 0.005 | 227.760 | 14.754 |
| 4500 | 0.005 | 229.294 | 14.809 |
| 5000 | 0.005 | 229.913 | 14.826 |
| 5500 | 0.005 | 230.593 | 14.814 |
| 6000 | 0.005 | 231.553 | 14.876 |
| 6500 | 0.005 | 232.121 | 14.885 |
| 7000 | 0.005 | 232.166 | 14.834 |
| 7500 | 0.005 | 232.416 | 14.835 |
| 8000 | 0.005 | 232.168 | 14.802 |
| 8500 | 0.005 | 232.064 | 14.793 |
| 9000 | 0.005 | 232.382 | 14.788 |
| 9500 | 0.005 | 232.094 | 14.750 |
| 10000 | 0.005 | 232.158 | 14.739 |
| 10500 | 0.005 | 231.861 | 14.698 |
| 11000 | 0.005 | 231.804 | 14.686 |
| 11500 | 0.005 | 231.672 | 14.672 |
| 12000 | 0.005 | 231.350 | 14.632 |

**Table 2.5.** Mean queuing time estimates for the WAIT, the IDLE, and the CLEAN queues, when the arrival is modelled with a negative exponential distribution with mean 15, and there are 8 **barmaids** in the system.

Two cases, discussed in Sections 2.5.1. and 2.5.2, can be identified "a-priori" for which methods for the elimination of the initialisation bias problem may not work or may not be required.

### 2.5.1. OSCILLATORY APPROACH TO THE STEADY STATE

Most queuing networks will have a monotonic, increasing or decreasing, approach to the steady state. The queues start empty or "idle" and then their queue length, and therefore their queuing time, increases or decreases until they reach the steady state.

However, there are other systems in which this approach to the steady state is oscillatory; this means that the mean estimates will oscillate around the steady state value, with oscillations getting smaller as the simulation run length increases. Here, use of a method for the elimination of the influence of the initial conditions does not work properly. In most methods the elimination of this influence is done by deleting some of the initial observations that are less representative of the steady state value. Doing this when the approach to the steady state is oscillatory will not eliminate the influence of the initial conditions. This is due to the fact that in this case there are local maximums followed by local minimums; the influence of the large values will be compensated by that of the small values and vice versa. Deletion of observations will not have a particular effect on how soon the steady state is reached (see analysis in Chapter 4, for the MILITARY model). Similar arguments are valid for other methods for the elimination of the influence of the initial conditions (see discussion in Chapters 4 and 6 for more details on this particular point).

### 2.5.2. SMALL CHANGE IN THE INITIAL STATE OF THE QUEUE

In some cases there is not a great change in the state of the queue and the initial conditions are similar to the steady state values. In other words, there is not a great change in the values of the mean estimates obtained for a very long simulation run length with respect to their value at the beginning of the simulation. In these cases, the steady state can be reached for very short simulation run lengths. Deletion of some of the initial observations is not only unnecessary, but a waste of computer time. In practice, these cases can be identified by running the simulation model for a long (but not very long) simulation run length, and observing the estimates obtained for the parameters of interest for a very short run length as well as for the long run length. This will give the simulation practitioner an idea of how much is the change in their values.

## 2.6. FURTHER CONSIDERATIONS

In the previous sections we have discussed some points that should be considered by the simulation practitioner before starting the simulation. This analysis is not always done but it is important and it may save time. In this section we mention some additional points that the simulation practitioner has to consider if the simulation is to give reliable results.

### 2.6.1. FURTHER DISCUSSION ON SMALL MEAN ESTIMATES

Sometimes, for example when there are several servers, the steady state mean queuing time and mean queue length of the queues belonging to temporary entities that interact with this server (like for example, the WAIT queue in the PUB model simulated in condition 3 in section 2.4.3., see mean estimates in table 2.5.) will be very small in value. In these cases, from a statistical point of view, we consider that other types of measurement of the "centre" of the distribution should be used instead of the sample mean. The reason for this is that the individual observations used by the simulation for the calculation of the mean estimate $X_i$ obtained in replication $i$ (assuming that the method of replications is used) are a combination of 0's and 1's and possibly 2's. This corresponds to a discrete distribution and then instead of estimating the mean we should estimate for example the proportion of time the queue is empty. As our main concern in this thesis has been to show how it is possible to develop simple procedures we have shown it using mean estimates. However, the use of other estimates than the sample mean in simulation requires further research.

### 2.6.2. DETERMINATION OF THE STEADY STATE

The main characteristic of this research is that it uses results obtained from the simulation of different models to propose new procedures, or to infer something about a particular problem of simulation. A second important characteristic is that using this approach we show that simulation procedures do not need to be difficult and theoretically based in order to give satisfactory answers to the questions of interest. It is shown that based on the observation of characteristics that are common to different simulation models, it is possible to formulate simple solutions for simulation problems that until now do not have a satisfactory answer. This empirical rather than theoretical approach may not always give the **optimal** answer, but a **good** one. However, as discussed in the Operational Research literature, sometimes the benefits obtained with the **optimal** solution do not justify the time and money spent on obtaining it.

For this reason, the steady state of the system is identified in this thesis not by the application of one of the sophisticated definitions of steady state, like for example, that in "the steady state the probability of each different state is known," but by a simple analysis of the simulation output data; as the simulation run length increases we can determine the run length for which the output data shows convergence to a value considered to be the steady state value.

It is important to notice that simulation theoreticians expect that a new proposed procedure will give estimates within 0.5% or even less from the real value $\mu$, and to show this, they test the procedure with simple systems like the M/M/1 queue. However, from a practical point of view, there is often no difference between a queue length of 1.25 or 1.28 (2.5% increase). Obviously the maximum tolerance (i.e., 2.5% or 1% or less) will depend on the particular problem that the simulation is trying to solve. Therefore, and considering that the research is not only based on results obtained from the simulation but that it tries to give some guidelines about the practical use of simulation, we consider for the purposes of this research that the parameters estimated in the different experiments are in a steady state if there is not a variation of more than 2.5% from the real steady state value.

In the simulation models used to obtain the results (like for example the PUB or the STEELWORKS) analysed in the thesis we do not know the steady state real value ($\mu$); however, by running the simulation model for extremely long simulation run lengths and using a very large number of replications we can obtain a good approximation for $\mu$ (Appendix C); this approach is not desirable in real life as it would greatly increase the cost of the simulation.

### 2.6.3. LARGE STANDARD DEVIATION AS COMPARED WITH THE SAMPLE MEAN

One of the problems faced by the simulation practitioner is that of a large standard deviation as compared with the sample mean. It occurs mainly in terminating simulations when the simulation run length is short. In some cases this may be due only to a phenomenon related to the random number seeds. Sometimes, for a particular combination of random number seeds and the input values to the specific model that is being simulated, outliers can appear in the simulation output. In these cases, the practitioner can identify the particular combination of random number seeds for which the problem appears and replace it by a different combination. However, in some other cases. the process itself has a large variance. Then, the confidence interval width will be very large and will not give an accurate idea of where the "centre" of the distribution is. This problem requires some special statistical treatment. It has been suggested to use the median instead of the mean to describe the output distribution as this statistic is less influenced by the outliers. It has also been suggested to use the trimmed mean that does not use the most extreme values of the sample for the calculation of the sample mean. In this case one of the disadvantages is that the statistical analysis of the results is not easy and may add to the complexity of the simulation. We have not used either of these approaches in this thesis as one of our main objectives was to show that simple procedures can work well in simulation, and we have done it using the sample means. An area of future research is the extension of the procedures proposed in this thesis to other parameters.

## 2.7. CONCLUSIONS

In this chapter we have given a more detailed picture of what this research is about, and created a scenario that is required for the reader to understand the approach followed in this research.

Emphasis has been put on the use of results obtained from the simulation of different models. It has also been discussed why the new proposed procedures do not have a rigorous mathematical justification. By doing this, simpler and easier to use procedures can be formulated; nevertheless, they give acceptable results at a smaller cost and greater simplicity and understandability especially for the user who has no practical experience in the application of simulation.

In order to show how simulation should be used in practice, we discussed in this chapter some aspects that should be studied by the practitioner before running the simulation. We also showed how this analysis prior to the simulation, although based on queuing theory for the particular examples discussed in this thesis, can be carried out by people with no special knowledge of it. In summary, the discussion in this chapter prepares the reader for the approach followed in the following chapters.

An important result of this chapter are the different guidelines given for the identification of possible **critical queues** that may apparently never reach a steady state or may require a very long simulation run length to reach it. In general, we show how the identification of possible unstable queues can be carried out in a simple way; we also show how an analysis of the system previous to running the simulation can help sometimes to identify those queues that may delay the system in reaching the steady state. However, as will be shown in Chapter 3, in those queues that may require a very long simulation run length to reach the steady state, an increase in the number of replications may reduce this required run length. This relationship between simulation run length required to reach the steady state and the number of replications, has not been discussed, to our knowledge, in the literature.

The message from this chapter is similar to what we can expect in the rest of the thesis: simulation does not require sophisticated and complex methods for its use and applications. Usually skill and understanding of the simulation model is more important than expensive and difficult to use software.

# CHAPTER 3 : ESTIMATION OF THE NUMBER OF REPLICATIONS REQUIRED IN A PARTICULAR SIMULATION

## 3.1. INTRODUCTION

One of the most important questions that the simulation practitioner has to answer before running the simulation is that of the number of replications to use. In terminating simulations, using a small number may give a non-accurate estimate. In steady state simulations we found, from empirical results, that using a larger number of replications permits the practitioner to detect clearly when the curve of the mean estimates as a function of the simulation run length becomes horizontal. This detection is not easy when the number of replications used is small because of the greater variability in the mean estimates; in this case the approach to the steady state is not as smooth as it is when we use more replications. Obviously when the variance of the estimate is smaller (and this corresponds to an increase in the number of replications) the mean estimate will be more accurate and will be closer in absolute value to the real steady state value. For example, in Table 2.3., the mean queuing time estimates for the WAIT queue did not fall within 2.5% of the steady state value for a simulation run length as long as 20000 (although it is not shown in the table, they require a simulation run length longer than 25000 minutes to reach the steady state). The data of this table was obtained from 100 replications. However, when 900 replications are used, the estimates fall within 2.5% of the steady state value for a simulation run length as short as 7000 minutes. Even if we consider that there is some variability in these results and that therefore, when different sets of random numbers are used these run lengths will be different, the mean estimates obtained for a larger number of replications will be more accurate than those obtained for a small number of replications.

As the method of replications is used in the following chapters, we considered it important to develop a method for the estimation of the number of replications to use in a given simulation; this method should be easy to apply, not time consuming and if possible should not require assumptions that sometimes are not met in practice. It is important to notice that when the method of replications is used for the estimation of steady state parameters, the first step is to decide on the number of replications to be used; the second step is to decide if the influence of the initial conditions is important and if so a method to deal with this problem should be used (See Chapters 4 and/or 6).

### 3.1.1. CHAPTER OBJECTIVES

Using some methods that have been proposed in the literature for the estimation of the number of replications we will develop in this chapter a method that can be used for the estimation of the number of replications to be used for the mean estimates to be sufficiently accurate; use of this number of replications will also allow the practitioner to detect the point in time for which the steady state is reached, which is equivalently to the point in time for which the curve of the mean estimates as a function of the simulation run length becomes horizontal.

### 3.1.2. CHAPTER OUTLINE

In Section 3.2. we give a brief discussion of some of the methods that have been proposed in the literature for the estimation of the sample size, or number of replications. Section 3.3. shows, from results obtained from the simulation, the influence of the number of replications on the curve of the mean estimates as a function of the simulation run length and in Section 3.4. we explain a method, based on those discussed in Section 3.2, and that can be used for the estimation of the number of replications to be used. In Section 3.5. we give empirical results for a simulation model for which no analytical answer can be obtained.

## 3.2. ESTIMATING A SINGLE MEAN : SAMPLE SIZE REQUIRED

In the method of replications, we make *k* replications, and use the average ($\bar{X}$) of the values $X_1$, $X_2$...$X_k$ obtained from the simulation as an estimate of the mean. However, when the variance of this mean estimate is too large we obtain inconclusive results, in the sense that the confidence interval calculated from these values is excessively large and will not give a clear idea of where the real value lies. The way of reducing this variance is by increasing the number of replications to $k1 > k$. "Sometimes practitioners "solve" the sample-size problem by continuing sampling until (say) the third digit after the decimal point does not change (that is, that digit does not change for the first time); of course such a procedure is not statistically sound." (Kleijnen, 1987). Therefore, there is a need for a procedure to guide the practitioner on how many replications to use.

This section describes some of the methods that have been proposed for the estimation of the number of replications required to give an "accurate" mean estimate. The methods discussed in section 3.2.1. are described by Kleijnen (1987) but similar methods can be found in other simulation publications (Law and Kelton, 1991; Banks and Carson, 1984; Law et al, 1981). In section 3.2.2. we discuss some of the problems associated with the use of these methods in practical simulation.

### 3.2.1. ESTIMATION OF THE NUMBER OF REPLICATIONS REQUIRED FOR THE ESTIMATION OF A SINGLE MEAN.

In this section we describe some of the methods currently employed for the estimation of the number of replications that are to be used in a simulation in order to obtain a confidence interval with a pre-specified width. This discussion has been taken from Kleijnen (1987)

"We begin with a simple (unrealistic) situation to illustrate sample-size determination. We assume a known variance (of course, we shall drop this assumption later) and a single normal population whose mean $\mu_t$ we wish to estimate. We want our estimate $\bar{X}$ to be less than (say) $c$ units wrong. Because of random noise we are never 100% certain of achieving this goal so that we settle for (say) 1-$\alpha$ certainty:

$$P(|\bar{X}-\mu| \leq c) = 1 - \alpha \tag{3.1}$$

The following relation holds (from basic statistic concepts):

$$P(|\bar{X}-\mu| \leq \frac{z^{\alpha/2}\sigma_x}{\sqrt{n}}) = 1 - \alpha \tag{3.2}$$

where $z^{\alpha/2}$ is the upper 1 - $\alpha/2$ critical point for a standard normal distribution, $\sigma_x$ is the standard deviation of the mean estimate and n is the sample size.

Hence if we want to meet both Eq. 3.1. and 3.2., then Eq. 3.3. must hold.

$$c = \frac{z^{\alpha/2}\sigma_x}{\sqrt{n}} \tag{3.3}$$

Consequently the sample size n should be:

$$n = (z^{\alpha/2}/c)^2\sigma_x^2 \tag{3.4}$$

In other words, the desired sample size increases as

1. The noise $\sigma_x$ increases.

2. The confidence interval width decreases (the "half-length" of the confidence interval is $c$).

3. The coverage probability increases (as 1 - $\alpha$ increases, $z^{\alpha/2}$ increases).

The size of the sample size reacts quadratically to these three factors." (Kleijnen, 1987)

In practice we have an unknown variance $\sigma_x^2$. Of course we can *estimate* $\sigma_x^2$ through the sample variance $s_x^2$.

When we replace $\sigma_x^2$ by $s_x^2$ then we replace $z^{\alpha/2}$ by $t_{n-1}^{\alpha/2}$ and then Eq. 3.4. transforms into:

$$n = (t_{n-1}^{\alpha/2}/c)^2 s_x^2 \qquad (3.5)$$

"However, strictly speaking, the last step resulting in Eq. 3.5. is false, because in Eq. 3.5. the sample size n has become a stochastic variable (n depends on the estimator $s_x^2$). In practice Eq. 3.5. often works, which is substantiated by a number of statistical studies. These studies prove analytically that when $s_x^2$ in Eq. 3.5. is updated after each additional observation, then for large sample sizes Eq. 3.5. is consistent and efficient. "Consistency" means that the confidence interval does indeed cover $\mu_t$ with prespecified probability $1 - \alpha$. "Efficiency" means that the *expected* sample size equals the sample size for known $\sigma_x^2$, given in Eq. 3.4. For small samples the consistency and efficiency do not change much, as is shown by Monte Carlo studies. (Anscombe, 1953; Chow and Robbins, 1965; Starr, 1966; Robbins et al, 1967; Srivastava, 1970). This is *purely sequential*; that is, we update $\sigma_x^2$ after each additional observation i' (i' = 2, 3...n)." (Kleijnen, 1987).

A slightly different approach is known as *double sequential.* In this approach we make initially $n = n_0$ replications. We calculate the corresponding standard deviation s(n) and the confidence interval *relative precision*, I, which is simply the confidence interval half-width divided by the mean estimate. This is mathematically expressed by Eq. 3.6.

$$I = \frac{t_{n-1}^{\alpha/2} s(n) \sqrt{(n)}}{\overline{X}} \qquad (3.6)$$

where $t_{n-1}^{\alpha/2}$ is the upper $1-\alpha/2$ critical point from the t-distribution, and n-1 degrees of freedom.

If $I \leq c'$, then we use $n = n_0$ replications. Otherwise, we increase n by 1, make an additional replication of the simulation, update the standard deviation and calculate I. We repeat this procedure until $I \leq c'$. In this context c' is a pre-assigned positive value chosen by the practitioner and that corresponds to the desired c.i. relative precision. Law and Kelton (1991) suggest to start the double-sequential approach with $n_0 = 10$ and use $c' \leq 0.15$.

"When we compare the purely sequential approach to the double-sample approach, we notice a practical problem: How should we choose the pilot sample size $n_0$? A large initial sample $n_0$ reduces $t_{n0-1}^{\alpha/2}$ (as long as $n_0$ is smaller than, say 30) and hence it reduces the total sample size. However, such a large pilot sample may result in wasted simulation runs, namely, if $n_0$ exceeds n. Large pilot samples tend to decrease the efficiency (large sample size) and to increase the consistency (high coverage probability)." (Kleijnen, 1987).

## 3.3. INFLUENCE OF THE NUMBER OF REPLICATIONS ON THE ACCURACY OF THE MEAN ESTIMATES OBTAINED FROM THE SIMULATION

In section 3.2. we discussed some of the methods that have been proposed in the literature for the estimation of the number of replications required for the estimation of a parameter. In this section we discuss two important aspects identified while studying this influence. Although it has not been discussed in the simulation literature, the random number streams used to obtain independent observations may have some influence on the accuracy of the mean estimate. This problem is discussed in section 3.3.1. A second aspect that is studied in section 3.3.2. is the influence of the number of replications on the accuracy of the curve of the mean estimates as a function of the simulation run length as compared to the real, but in general unknown, real curve.

### 3.3.1. INFLUENCE OF THE RANDOM NUMBER STREAMS ON THE ESTIMATES OBTAINED FROM A SIMULATION

We will discuss in this section the influence of the random number streams on the mean estimates obtained from the simulation. We found that in practical simulations, when a "small" number of replications is used, the mean estimate obtained from the simulation for short simulation run lengths will or may depend on the set of random number streams used to obtain *k* independent observations. To show the influence of the random number streams on the mean estimates, we simulated the LAUNDERETTE model and obtained mean estimates for the WASHQ mean queuing time as a function of the simulation run length. The mean estimates were obtained from 100 independent replications, where the set R.N.[1] = $[R_1, R_2,...R_{100}]$ of random number streams was used. $R_i$ are different and independent random number streams. The experiment was repeated for a different set of random number streams R.N.[2] = $[R_{101}, R_{102},...R_{200}]$.

Figure 3.1. shows the mean estimates as a function of the simulation run length for the two different sets of random number streams. It can be seen how the mean estimates differ greatly, especially for short simulation run lengths.

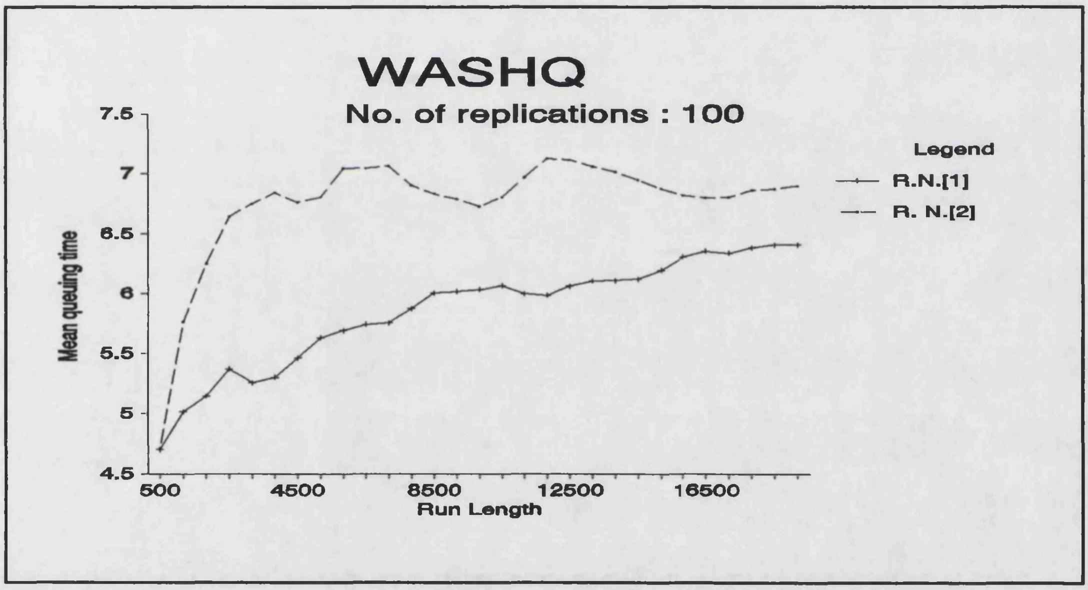



**Figure 3.1.** WASHQ mean queuing time estimates as a function of the simulation run length, for two different sets of random number streams and 100 replications.

This influence of the random number streams is due to the presence of extreme values, very large or very small, in the observations of some of the replications for one of the sets of the random number streams. These extreme values do not appear in the set of observations obtained for a different set of random number streams. This influence tends to disappear when the number of replications is increased, as in this case the influence of extreme values is less, and they do not have weight as large as when the mean estimate is the average of fewer observations. This is shown in Figure 3.2. which shows the WASHQ mean queuing time estimates obtained from 900 replications.

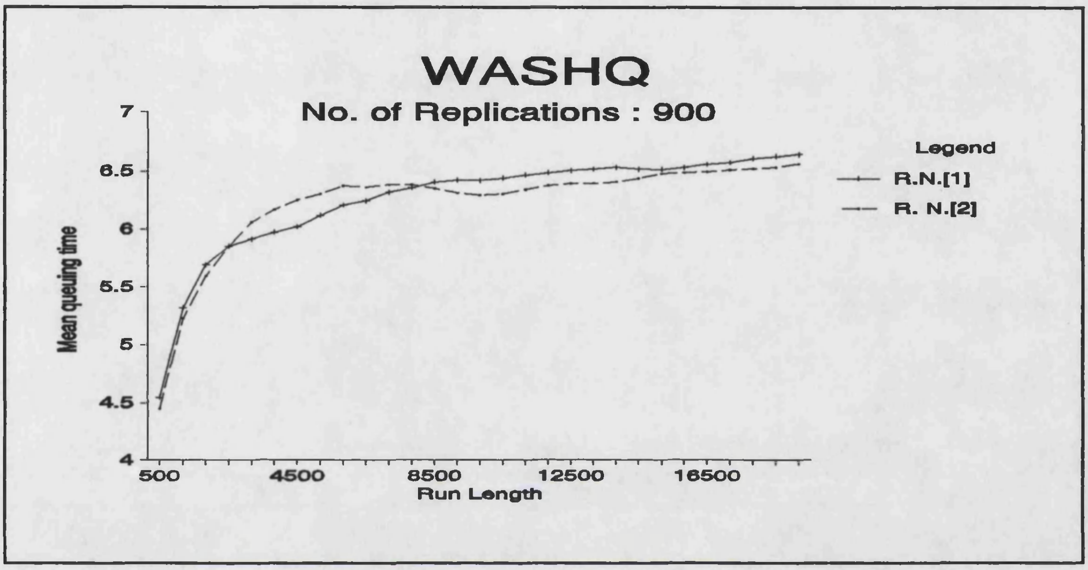



**Figure 3.2.** WASHQ mean queuing time estimates as a function of the simulation run length for two different sets of random number streams, and 900 replications.

### 3.3.2. NUMBER OF REPLICATIONS AND SIMULATION RUN LENGTH REQUIRED TO OBTAIN A GOOD ESTIMATE OF THE STEADY STATE

In Section 3.3.1. we discussed how when the number of replications $k$ is small, the random number streams used to obtain the $k$ independent observations can have an important influence on the mean estimate obtained from the simulation, especially for short simulation run lengths.

We also showed how when the number of replications is increased the influence of the random number streams is negligible. A second important aspect of the influence of the number of replications is that the curve of the mean estimates as a function of the simulation run length becomes closer (i.e., is a better approximation) to the real, but unknown curve, as the number of replications increases. This in practical terms implies that the approach to the steady state is smoother and that it is easier to estimate the simulation run length for which the curve becomes horizontal when a larger number of replications are used. This is illustrated in Figure 3.3. In this figure we show the WASHQ mean queuing time estimates as a function of the simulation run length and of the number of replications. We show the results obtained for 100 replications and two different sets of random number streams and the results for 900 replications. In this last case the results are similar independent of the random number streams. This graph shows how the approach to the steady state is smoother and quicker if 900 replications are used.

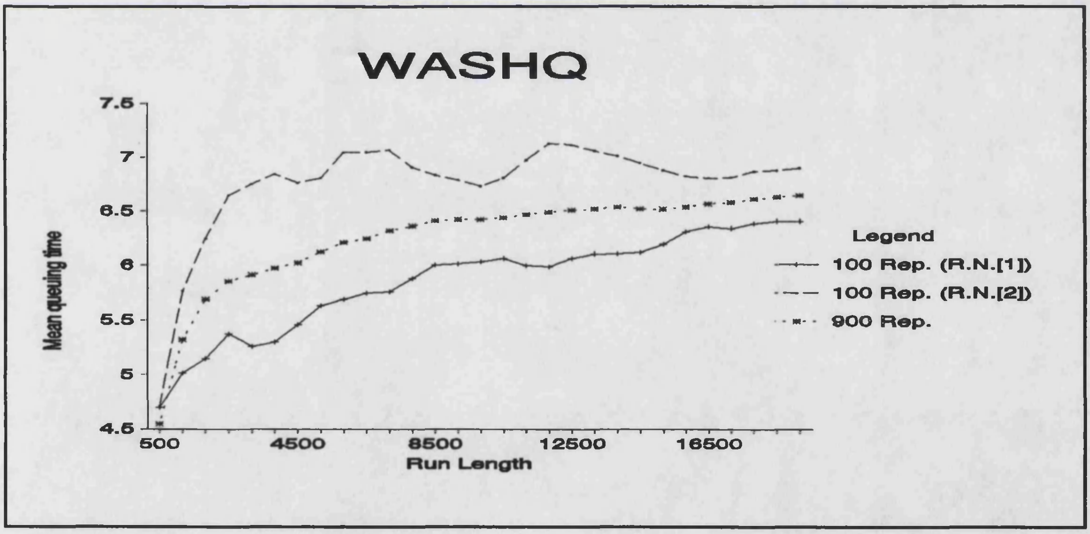



**Figure 3.3.** WASHQ mean queuing time estimates as a function of the simulation run length, of the number of replications, and of the set of random number seeds.

## 3.4. ESTIMATION OF THE NUMBER OF REPLICATIONS REQUIRED FOR THE ACCURATE ESTIMATION OF A PARAMETER IN SIMULATION

In Section 3.3. we identified two important and practical aspects of the influence of the number of replications on the mean estimates: 1. The influence of the random number streams on the accuracy of the mean estimates and 2. The influence of the number of replications on the shape of the curve of the mean estimates as a function of the simulation run length.

These two aspects were the motivation for the present study. To overcome these problems we want to develop a simple method for the estimation of the number $k$ of replications. We will show how, once this number $k$ has been estimated, it can be used in any simulation of this model, for this particular parameter, independent of the simulation run length.

This number, if correctly estimated, will give estimates that are independent of the random number streams. At the same time, because once the influence of the random number streams becomes negligible, the mean estimates do not differ too much, this number of replications will give accurate mean estimates for both terminating and steady state simulations.

### 3.4.1. PROPOSED METHOD FOR THE ESTIMATION OF THE NUMBER $K$ OF REPLICATIONS

The methods discussed in section 3.2. that have been proposed for the estimation of the number of replications, or of observations to record in a given simulation have been extensively tested, and they seem to perform well when applied to simple models for which an analytical answer can be calculated. By applying the sequential approach described in Section 3.2. to simulation models for which no analytical answer can be calculated we will show that they can be used for the estimation of the number of replications that is required in order to obtain a good approximation to the real, but unknown, curve of the mean values as a function of the simulation run length.

Our objective is to estimate a GOOD, but not the OPTIMAL, number of replications such that this curve will be a close approximation of the real one.

This number of replications can be estimated using the double sequential approach described in Section 3.2.1. This can be done for a short simulation run length and the value thus estimated can be used then to obtain accurate steady state parameters. Some points need to be discussed:

1. **What "short simulation run length" means.** We found from applying the method to different simulation models for which no analytical answer can be obtained that "short" depends on the time that the different activities take to be executed. If among the mean values of these execution times for the different activities the maximum takes $D_0$ units of time, a "short" simulation run length can be 10 to 15 times the value of $D_0$.

2. **Which value should be used for "c'".** The value of c' can be chosen to be 0.05 or less as we are using a short simulation run length for the estimation of the number of replications to be used in steady state simulations, and the estimates for short simulation run lengths tend to have larger variance.

3. **Number N of independent estimates.** Although Law and Kelton (1991) suggest to start the double sequential approach using $n_0 = 10$, we found that in more typical simulation models $n_0$ should be larger. We chose $n_0 = 100$. Another advantage of choosing $n_0 \geq 100$ is that then the central limit theorem will guarantee that the distribution of the mean values follow a normal distribution.

In summary the proposed procedure is as follows: select a short simulation run length (obviously there will still be some influence of the transient) and make at least 100 replications. Based on the results obtained from these replications, and following the double sequential approach, estimate the number n of replications to be used for the estimation of steady state parameters. This number n will make it easier to estimate the point in simulated time for which the curve of the mean estimates as a function of the simulated time becomes horizontal.

This procedure was applied to three queues of the PUB simulation model and the results are given and discussed in Section 3.5.

## 3.5. ANALYSIS OF THE EMPIRICAL RESULTS

Section 3.4.1. proposed a method that can be used to obtain an estimate of the number of replications that are required when the objective of the simulation is the estimation of steady state parameters. In this section we will illustrate the use of this algorithm by applying it to some queues of the PUB model. Additional examples are given in Appendix D.

Our main objective in the example given in this section is to show how, using the number of replications estimated following the procedure of Section 3.4.1., we obtain a curve of the mean estimates as a function of the simulation run length which is a good approximation to the real, but unknown, one. Therefore, it will be easier to estimate the simulated time for which the curve becomes horizontal if this number of replications is used than if less replications are used. In order to show this we will divide the following study into two parts:

1. The estimation of the number of replications and

2. The evaluation of the performance of the proposed procedure. We expect that the graph of the mean estimates as a function of the simulation run length will be "smoother" when the estimated number of replications is used than when less replications are used.

### 3.5.1. ESTIMATION OF THE NUMBER OF REPLICATIONS

Three queues of the PUB model (See Figure A.4., Appendix A) have been studied: WAIT, CLEAN and IDLE. For each one of these queues and for the mean queuing time parameter, we want to estimate the number of replications that should be used for the estimation of accurate steady state parameters. For the estimation of this number of replications we chose a short simulation run length (500 minutes) and obtained mean estimates for 100 replications.

Following the procedure described in Section 3.4.1. we used a value of $c' = 0.025$, and obtained the following values for n:

| QUEUE | Value of n |
|---|---|
| WAIT | 850 |
| CLEAN | 160 |
| IDLE | 550 |

### 3.5.2. EVALUATION OF THE PERFORMANCE OF THE PROPOSED PROCEDURE.

In this section we will show how, when the number of replications estimated above are used, the graph of the mean estimates as a function of the simulation run length is "smoother" and it is easier to estimate the point in simulated time for which the curve becomes horizontal.

To show this we will obtain the mean estimates as a function of the simulation run length and of different number of replications and we will show the results on a graph. From this graph we should be able to approximate the point for which the parameter seems to have reached the steady state. We will then obtain c.i. for longer simulation run lengths and compare them in terms of their width, and of their coverage (i.e, do they "cover" the real value which has been estimated in Appendix C, or not). The real steady state values which have been estimated in Appendix C are the following:

| Queue | Steady state ($\mu$) |
|---|---|
| WAIT | 1.140 |
| CLEAN | 209.400 |
| IDLE | 2.001 |

Figures 3.4., 3.5., and 3.6. give the mean queuing time estimates of the WAIT, CLEAN and IDLE queues respectively as a function of the simulation run length and of the number of replications. From these figures it is easier to estimate the simulation run length for which the curve of the mean estimates becomes horizontal if we use the number of replications estimated following the procedure of section 3.4.1. than if we use less replications.

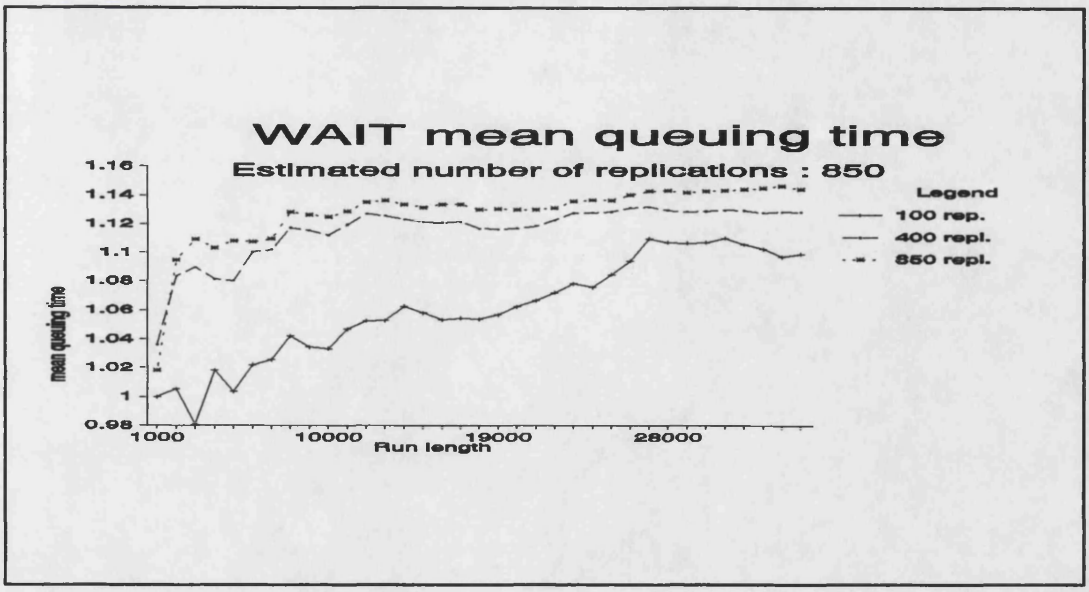



**Figure 3.4.** WAIT mean queuing time estimates as a function of the number of replications and of the simulation run length.

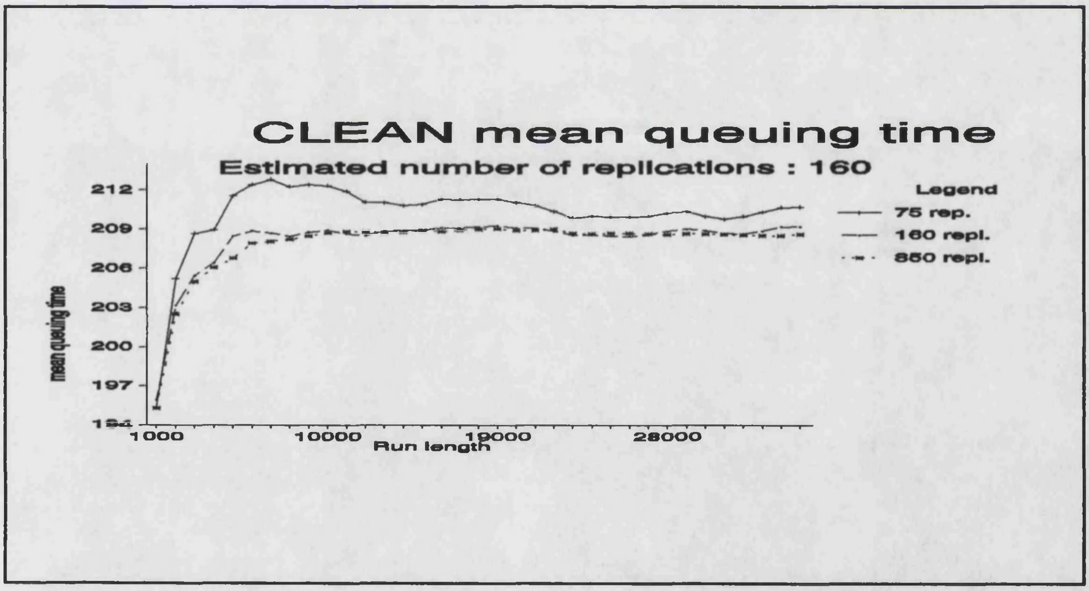



**Figure 3.5.** CLEAN mean queuing time estimates as a function of the number of replications and of the simulation run length.

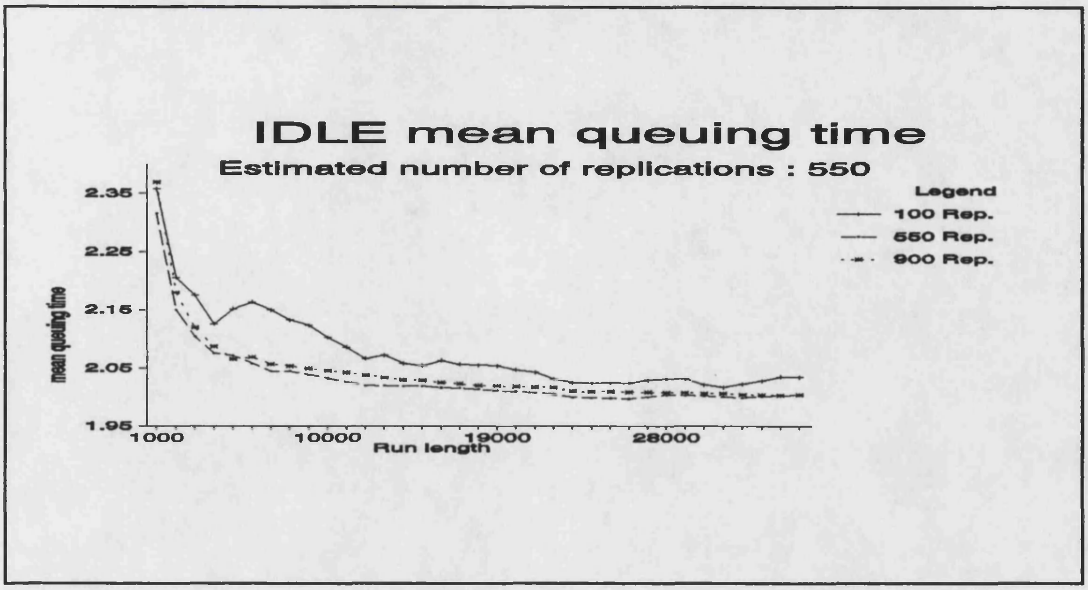



**Figure 3.6.** IDLE mean queuing time estimates as a function of the simulation run length and of the number of replications.

For example, from figure 3.4. we can notice how the curve becomes almost horizontal for a simulation run length of 7000. This is noticeable when at least 400 replications are used, although the estimation is more accurate when 850 replications are made. Similarly, from Figure 3.6. how the curve becomes almost horizontal for a simulation run length of 4000. In this case we require 550 replications. The figures also show how an increase in the number of replications does not make an important difference on the smoothness of the curve.

To show how the number of replications affect the c.i. obtained, tables 3.1., 3.2., and 3.3. give the c.i. half-width, the c.i. lower and upper limit (using 95% c.i.) and the percentage error for the WAIT, CLEAN, and IDLE mean queuing time, for different run lengths and different number of replications.

Although results are not given here, when 100 independent c.i. were calculated based on (only) 100 replications for the run lengths approximated from the graphs and for which the curve becomes almost horizontal, the coverage was close to $100(1 - \alpha)\%$ as expected.

| Results based on 100 observations | | | | | | |
|---|---|---|---|---|---|---|
| Run | mean | std.Dev. | c.i.half-w | c.i.l.l. | c.i.u.l. | % error |
| 7000 | 1.1 | 0.5 | 0.09 | 1.02 | 1.21 | 2.31 |
| 19000 | 1.1 | 0.3 | 0.06 | 1.07 | 1.18 | 1.75 |
| 27000 | 1.1 | 0.3 | 0.05 | 1.09 | 1.20 | -0.13 |
| Results based on 400 observations | | | | | | |
| Run | mean | std.Dev. | c.i.half-w | c.i.l.l. | c.i.u.l. | % error |
| 7000 | 1.1 | 0.5 | 0.05 | 1.07 | 1.17 | 1.57 |
| 19000 | 1.1 | 0.3 | 0.03 | 1.11 | 1.17 | 0.07 |
| 27000 | 1.2 | 0.3 | 0.03 | 1.12 | 1.18 | -1.00 |
| Results based on 900 observations | | | | | | |
| Run | mean | std.Dev. | c.i.half-w | c.i.l.l. | c.i.u.l. | % error |
| 7000 | 1.1 | 0.5 | 0.03 | 1.08 | 1.14 | 2.43 |
| 19000 | 1.1 | 0.3 | 0.02 | 1.10 | 1.14 | 1.46 |
| 27000 | 1.1 | 0.3 | 0.02 | 1.13 | 1.17 | -0.47 |
| Results based on 1200 observations | | | | | | |
| Run | mean | std.Dev. | c.i.half-w | c.i.l.l. | c.i.u.l. | % error |
| 7000 | 1.1 | 0.4 | 0.03 | 1.09 | 1.14 | 2.42 |
| 19000 | 1.1 | 0.3 | 0.02 | 1.11 | 1.14 | 1.38 |
| 27000 | 1.1 | 0.3 | 0.02 | 1.13 | 1.16 | -0.12 |

**Table 3.1.** WAIT mean queuing time c.i. half width, c.i. upper and lower limit (c.i.l.l. and c.i.u.l.) and percentage error of the mean estimate as compared to the real steady state value.

| Results based on 100 observations | | | | | | |
|---|---|---|---|---|---|---|
| Run | mean | std.Dev. | c.i.half-w | c.i.l.l. | c.i.u.l. | % error |
| 7000 | 208.8 | 18.7 | 3.71 | 205.1 | 212.5 | 0.3 |
| 19000 | 210.2 | 11.7 | 2.33 | 207.9 | 212.5 | -0.4 |
| 27000 | 210.0 | 10.6 | 2.10 | 207.9 | 212.1 | -0.3 |
| Results based on 200 observations | | | | | | |
| Run | mean | std.Dev. | c.i.half-w | c.i.l.l. | c.i.u.l. | % error |
| 7000 | 209.0 | 18.5 | 2.57 | 206.4 | 211.6 | 0.2 |
| 19000 | 209.2 | 11.6 | 1.61 | 207.6 | 210.8 | 0.1 |
| 27000 | 208.6 | 10.7 | 1.48 | 207.2 | 210.1 | 0.4 |
| Results based on 400 observations | | | | | | |
| Run | mean | std.Dev. | c.i.half-w | c.i.l.l. | c.i.u.l. | % error |
| 7000 | 208.3 | 19.5 | 1.92 | 206.4 | 210.2 | 0.5 |
| 19000 | 209.4 | 11.9 | 1.17 | 208.2 | 210.6 | 0.0 |
| 27000 | 209.0 | 10.8 | 1.05 | 208.0 | 210.1 | 0.2 |
| Results based on 1200 observations | | | | | | |
| Run | mean | std.Dev. | c.i.half-w | c.i.l.l. | c.i.u.l. | % error |
| 7000 | 207.6 | 19.0 | 1.08 | 206.5 | 208.7 | 0.9 |
| 19000 | 209.2 | 11.8 | 0.67 | 208.5 | 209.9 | 0.1 |
| 27000 | 208.8 | 10.3 | 0.58 | 208.2 | 209.4 | 0.3 |

**Table 3.2.** CLEAN mean queuing time c.i. half width, c.i. upper and lower limit (c.i.l.l. and c.i.u.l.) and percentage error of the mean estimate as compared to the real steady state value.

| Results based on 100 observations | | | | | | |
|---|---|---|---|---|---|---|
| Run | mean | std.Dev. | c.i.half-w | c.i.l.l. | c.i.u.l. | % error |
| 7000 | 2.1 | 0.4 | 0.08 | 2.00 | 2.15 | -3.8 |
| 19000 | 2.0 | 0.2 | 0.05 | 1.99 | 2.09 | -2.0 |
| 27000 | 2.0 | 0.2 | 0.04 | 2.00 | 2.08 | -2.0 |
| Results based on 400 observations | | | | | | |
| Run | mean | std.Dev. | c.i.half-w | c.i.l.l. | c.i.u.l. | % error |
| 7000 | 2.1 | 0.4 | 0.04 | 2.03 | 2.11 | -3.4 |
| 19000 | 2.0 | 0.2 | 0.02 | 2.01 | 2.06 | -1.7 |
| 27000 | 2.0 | 0.2 | 0.02 | 2.00 | 2.04 | -1.1 |
| Results based on 700 observations | | | | | | |
| Run | mean | std.Dev. | c.i.half-w | c.i.l.l. | c.i.u.l. | % error |
| 7000 | 2.1 | 0.4 | 0.03 | 2.02 | 2.08 | -2.5 |
| 19000 | 2.0 | 0.2 | 0.02 | 2.02 | 2.05 | -1.8 |
| 27000 | 2.0 | 0.2 | 0.01 | 2.01 | 2.04 | -1.2 |
| Results based on 1200 observations | | | | | | |
| Run | mean | std.Dev. | c.i.half-w | c.i.l.l. | c.i.u.l. | % error |
| 7000 | 2.1 | 0.4 | 0.02 | 2.03 | 2.07 | -2.5 |
| 19000 | 2.0 | 0.2 | 0.01 | 2.01 | 2.04 | -1.2 |
| 27000 | 2.0 | 0.2 | 0.01 | 2.00 | 2.02 | -0.6 |

**Table 3.3.** IDLE mean queuing time c.i. half width, c.i. upper and lower limit (c.i.l.l. and c.i.u.l.) and percentage error of the mean estimate as compared to the real steady state value.

As can also be seen from the tables, and this conclusion is valid when several independent c.i. are calculated, another advantage of obtaining more replications is that the mean estimate becomes more accurate. This can be seen from the values of the c.i. half-width (or half-length). In general, "if the c.i. half-length is less than or equal to ß (where ß > 0) then:

$$1 - \alpha \approx P(\bar{X} - \text{half length} \le \mu \le \bar{X} + \text{half length})$$
$$= P(|\bar{X} - \mu| \le \text{half-length})$$
$$\le P(|\bar{X} - \mu| \le \beta)$$

" (Law and Kelton, 1991)

Obviously, and this is confirmed in the tables, as the number of replications increases the value of the c.i. half-length decreases, and therefore, we can expect a smaller absolute error in the value of the mean estimates.

Additional examples are given in Appendix D that confirm the results obtained here.

## 3.6. CONCLUSIONS

One important aspect not discussed before in the literature has been identified in this chapter: the influence of this number of replications on the estimation of the simulation run length required for a parameter to reach the steady state.

These two points led to the proposal of a simple method for the estimation of the number of replications for which the curve of the mean estimates as a function of the simulation run length is a good and close approximation to the real, but unknown, one. This allows the practitioner to estimate the simulated for which the curve becomes almost horizontal, i.e., for which the parameter can be considered to reach the steady state. It is important to obtain a good estimate of this point because as discussed by Law (1977) if the simulation run length is not long enough for the influence of the initial conditions to have disappeared, then the mean estimate will be biased, and in fact if we increase the number of replications the c.i. coverage (i.e, the percentage of c.i. covering the real steady state value) will tend to zero as a result of the c.i.'s being built not around the steady state value, but around a transient value.

In any case it is important to note that although the first step in a simulation, when the number of replications is used, is to estimate this number, the next step should be to deal with the influence of the initial conditions, especially if this influence is strong. In other words, the results of this chapter should not be used on their own, because it is clear that in the method of replications we are always starting the simulation with the same conditions which are usually not representative of the steady state conditions. However, the analysis carried out in this chapter provides the simulation practitioner with a starting point.

It may also be argued that instead of using a large number of replications it is possible to increase the simulation run length. This is valid, but it has the problem that the simulation run length is another parameter chosen a priori by the practitioner and it tends to be parameter and model dependent. And as shown in the graphs of this chapter when few replications are made the estimation of the simulation run length required for the parameter to reach the steady state is not easy.

We can think of this chapter as an extension to the topic of **critical queues** discussed in Chapter 2. In those queues identified as critical use of a larger number of replications may help to obtain a more accurate estimate.

# CHAPTER 4 : STEADY STATE : REDUCING THE TRANSIENT PHASE

## 4.1. INTRODUCTION

Textbooks, articles, conferences and symposiums all discuss the problem of the influence of the initial conditions. As discussed in Section 4.2. the behaviour of the *transient* is not representative of the steady state of the system. One of the most common ways of dealing with this problem is by deleting some of the initial observations which usually will greatly differ from the steady state values.

In one of the most complete reviews of the methods found in the literature to deal with the problem, Gafarian et al (1978) showed, based on five performance measures defined below, that none of the methods that existed at that time performed well. (See also, Wilson and Pritsker (1978a)).

The following notation is used to explain these different performance measures:

$\{X_t\}$: Stochastic process with index (time) parameter t.

L : Truncation point for a time series realisation of $\{X_t\}$;

$\mu_x$ : Steady State mean of the process $\{X_t\}$.

$\epsilon$ : preassigned relative tolerance.

We define t* as the minimum time such that:

$1 - \epsilon \leq E(X_t)/\mu_x \leq 1 + \epsilon$ for all $t \geq t^*$

Choosing $\epsilon$ sufficiently small, the stochastic process $\{X_t\}$ can be said to be in the steady state for values of $t \geq t^*$; this means that the expected value $X_t$ is close to $\mu_x$ and $E\{X_t\}/\mu_x$ is close to 1.

Depending on the procedure and the random number seeds used, the value of *L* varies and, therefore, we can think of *L* as a random variable used to estimate t*. Gafarian et al compare the different procedures based on the following measures:

1. *Accuracy*: $a = E(L)/t^*$ should be equal to one.

2. *Precision* : $p = \sigma(L)/E(L)$ should be close to zero.

3. *Generality*: A truncation point should perform well for a broad range of models.

4. *Cost*: Because some observations are being deleted, the computer time taken to do this should not be excessive.

5. *Simplicity*: The proposed procedure should be easy to understand and easy to use.

When applying these performance measures, some of the procedures studied were found to underestimate the truncation point, with the influence of the initial conditions remaining and leading to a biased steady state estimate. Other procedures were found to overestimate the truncation point leading to a waste of resources.

We now discuss some of these performance measures. With respect to the *cost* we must notice that when this study was carried out (1978) the cost of computer time could greatly increase the total cost of the project. With the development of new and faster technologies, cost is not as much a problem as it was some years ago. However, if the computer time spent in the estimation of the number of observations to be deleted is kept as small as possible, the practitioner can spend more time in taking a larger number of replications and/or increasing the total simulation run length. These two factors affect the accuracy of the estimate obtained from the simulation, and the larger they are the more accurate the estimate is.

If cost is not an important factor in the evaluation of a procedure proposed for the elimination of the influence of the initial condition, measures of performance 3 (*generality*) and 5 (*simplicity*) are still very important. Most of the procedures that have been proposed to deal with the initialisation bias problem are not shown to be *general*; they have used simple models, like the M/M/1 queue, to show the results. However, complex systems have a completely different behaviour. Therefore, the methods so far tested may not be expected to work satisfactorily for different types of real and complex systems. A second problem with these methods is that some of them, as is shown in section 4.3, are not easily implemented and for complex models they may require several computations, which will increase the time taken by the project (see comment in Section 1.1.2. about simulation projects taking an excessively long time). And, due to their complexity, most of these methods do not meet the fifth performance measure of *simplicity*. Two methods (Law and Kelton's and Welch's) are described in section 4.3. to illustrate these three problems (long time taken to finish the project due to complexity, lack of generality and lack of simplicity). The description of these two methods in this chapter will show how relatively easy it is to implement the method proposed in this chapter, and its generality is illustrated by applying it to simple and complex models.

Some other problems associated with the deletion of observations as a way of dealing with the initialisation bias problem have been discussed in the literature. For example, some authors question the usefulness of this method and they consider that deleting the transient part of a time series may give a biased variance and steady state estimate (Deutsch, et al; 1983). Some other authors have shown (based on simple models like the M/M/1 queue) that the deletion of some observations will increase the variance of the steady state estimate. However, new sampling methods have been proposed recently that may reduce the standard deviation of the mean estimate by up to 50% in some cases (Saliby, 1990a, 1990b). Law (1977), comparing the batch means with the replications method, showed that deletion can increase the bias when the simulation run length is not sufficiently large. Blomqvist (1970) showed that for the M/M/1 queue and some other simple queues, the mean squared error of the estimate increases with the deletion of some of the initial observations. This mean squared error is defined as $E[X(m,L)-\mu]^2$, where X(m,L) is the steady state estimate when $m$ observations are recorded but, of these $m$ observations, $L$ are deleted. "$\mu$" is the real, and (usually) unknown value. More generally, it has been shown that for a first-order autoregressive process, the mean-squared error may increase or decrease depending on the values of m and L. (Fishman (1972), Turnquist and Sussman (1977), Wilson and Pritsker (1978a), Snell and Schruben (1979), and Kelton and Law (1984)).

However, further research is needed in this field before rejecting the usefulness of the deletion of some of the initial observations, as most of these studies are based on observations made on the M/M/1 queue or other similar simple models.

### 4.1.1. CHAPTER OBJECTIVES

As mentioned above, the existing methods to deal with the initialisation bias problem do not perform well in practice, or are not easy to employ and to apply. Therefore, the objective of this chapter is to seek to develop a method *for the selection of a truncation point and which at the same time is easy to implement and to understand by the user with no previous special knowledge of the field.*

We do not try to develop a method supported by a rigorous mathematical theory as is usually done, but to use an empirical approach for it. By analysing different types of simulation models and their behaviour for short simulation run lengths we were able to identify a common pattern which could be due to the influence of the initial conditions. Further experimentation confirmed this point. In this way our method estimates the truncation point using results obtained from the simulation. The main advantage of this approach is that it does not require complex computations or modifications to the simulation software that are difficult to implement.

### 4.1.2. CHAPTER OUTLINE

Section 4.2. defines the term **transient phase** and discusses the problem of **Initialisation Bias**, as a consequence of the transient phase. Section 4.3. describes some of the methods that have been developed to deal with the problem of the transient phase in the calculation of steady state parameters. Section 4.4 develops a method to deal with the *initialisation bias* problem. This method in its present form is applied to queuing networks. An area for future research could be its extension so that it can be applied to other types of simulation models. The method was applied to simulation models for which no analytical answer can be calculated and Section 4.5. discusses some of the empirical results thus obtained. However, to test if the proposed method has some limitations, or if it works for simple models as well as it does for complex ones, it was also applied to three systems with known analytical steady state values and which are favourites amongst simulation theoreticians; these systems are the M/M/1 queue, the M/M/4 queue and a 2-stage queuing system (Queues in tandem).

## 4.2. TRANSIENT PHASE :WHAT IT IS

"A simulated system is considered to be in a steady state if its current behaviour is independent of the starting conditions, and if the probability of being in one of its states is governed by a fixed probability function. This does not mean that the system does not change state, but that the probability of being in any of its possible states can be determined." (Pidd, 1992).

The *transient phase* is the period of time between the start of the simulation and the final or steady state. This is better explained with an example. Suppose that a factory starts its operation. In one of its departments there is an assembly line where each shift picks up where the previous one left off. The first few hours or days of operation will not be really representative of the behaviour of the assembly line. This is due to the fact that the "queues" (points of the assembly line where the product has to wait for an operator to start working on it) will be initially "empty" and the workers will be initially "idle". This means that the first few elements to go through the assembly line will not have to wait in a queue anywhere, and just the same, some of the operators will have to wait longer (will be idle) before starting to work. However, as time increases, the flow in the assembly line will tend to stabilise and it will be possible to answer questions like how long it takes on the average for a product to go through the assembly line, and what percentage of time an operator is "idle" in a typical shift, etc. The period of time until the operation of the assembly line is "stabilised" is called the *transient phase* of the assembly line. Just the same, if a change is made in the assembly line, for example by reducing the number of operators, there will be another *transient phase* while the system readjusts to the new operating conditions. The important point here is that the operation of the system during the transient phase may be of little consequence when compared to the operation once a steady state has been reached.

### 4.2.1. DELETION OF SOME OF THE INITIAL OBSERVATIONS DURING THE TRANSIENT PHASE

While in some cases we are interested in the *transient phase* (for example, in terminating simulations), in other cases, when the main objective of the simulation is the study of the behaviour of the system in the long run, i.e., when we are interested in the estimation of steady state parameters, the influence of the transient phase may lead to biased estimates. This is due to the fact that the values of the observations during this transient phase are (or may be) quite different from the steady state values, and therefore using these values will give a sample mean either too small (when at the beginning of the simulation the queue is "empty") or too large (all entities in the queue are initially "idle") as compared to the real, but unknown, steady state values. Obviously, this could be overcome if the system is simulated for an extremely long period of time but this procedure will take a long continuous computer operation and increase the time taken by the project. Two possible solutions have been suggested:

1. Start the simulation with conditions that are more similar to the steady state conditions (Madansky, 1976). The problem with this approach is that this steady state is not known. In most simulations "the simplest course open to the analyst is to begin the simulation with no activity occurring and with the queues empty" (Pidd, 1992)

2. Do not record the initial output from the simulation. In other words, divide the total simulation time in two periods: an initial period $T_2$ and a second period $T_3$, with $T_3 >> T_2$. Do not take any record of the simulation

output data during time period $T_2$. (See Figure 4.1)

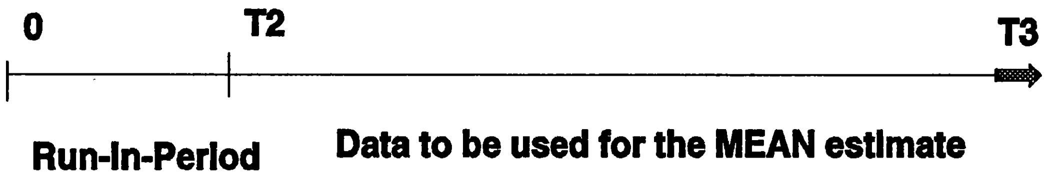



**Figure 4.1. "Warm-Up" + Data Collection Period**

This means that for a period of time $T_2$ we are driving the system into conditions which are more representative of the steady state and during this period we are not recording any data as this is quite different from the steady state values. For practical purposes, due to the type of simulation software used in the experiments carried out in this research, this period of time $T_2$ will be called the *Run-in-period*; in practice, simulation output is recorded from the beginning of the simulation run, but when the simulated time gets equal to the run-in-period, all data collected up to that point is discarded.

The basic problem now is to determine the length of $T_2$. If it is too small, the estimate will be biased; if it is too large, the variance of the estimate will increase (if the total simulation run length is not increased) and computer time will be wasted. Several methods have been developed to calculate this period of time.

Some methods, delete a fixed number of observations L in each replications, instead of deleting the observations recorded during a fixed period of time in each replication. When the first L observations of each replications are deleted, the period of time required to record them is a random variable that takes different values (although similar) in each replication. On the other hand, when a fixed period of time is chosen and the observations recorded during this period of time are deleted, the number of observations deleted is a random variable that takes different values in each replication. In the methods described in Section 4.3. the truncation point is given by a fixed number of initial observations L that are deleted in each replication. Four of the methods that have been proposed in the literature for the estimation of the number L of observations to be deleted have been compared by Kimbler and Knight (1985); it is interesting to notice that although the methods seem to perform well, at least in the case used to compare them, there is "quite a difference in the truncation points given by the various methods." This implies that in some cases the **truncation point** is overestimated and its overestimation is undesirable because the variance of the estimate increases. To explain this in more detail, if we call $L$ the number of observations deleted, and $m$ the total number of observations recorded from the simulation run, the estimate will be obtained from $m - L$ observations. As the variance of the estimate is inversely proportional to the number of observations used to calculate the mean estimate, if we keep $m$ fixed the larger the value of L the larger the variance of the mean estimate.

Summarising this section, the initial conditions are not representative of the steady state. Therefore they may produce a biased estimator. A possible solution which is further discussed in the remainder of this chapter is to run the system for an initial period of time called the run-in-period; any data recorded during this period of time is discarded. Doing this transforms the problem into that of how to obtain a "good value" for this initial period of time.

## 4.3. SOME METHODS FOR THE ELIMINATION OF INITIALISATION BIAS IN SIMULATION

Two methods that have been proposed in the literature for dealing with the initialisation bias problem are described in this section. This will give an idea of the complexity of the methods that are supported by a strong mathematical background. The first method, described in sections 4.3.1. and 4.3.2., is Kelton and Law's method (Kelton, 1980, 1982, and Kelton and Law, 1983). The second method is Welch's method (1983) and it is described in Section 4.3.3.

### 4.3.1. KELTON AND LAW'S METHOD

In this method, *k* replications of length *m* (i.e. m observations are recorded in each replication) are obtained. To reduce the influence of the initial conditions, the first *L* observations of each replication are deleted. The sample mean for replication *j* is given by equation 4.1. where $X_i(j)$ is the ith. observation obtained in the jth. replication.

$$\bar{X}_{Lm}(j) = \frac{1}{(m-L)}\sum_{i=L+1}^{m} X_i(j) \quad j=1,2...k \tag{4.1}$$

Using these values, an estimate of the mean $\mu$ is then given by equation 4.2.

$$\bar{X} = \frac{1}{k}\sum_{j=1}^{k}\bar{X}_{Lm}(j) \tag{4.2}$$

The method tries to find a <u>good</u> but not necessarily <u>optimal</u> value for *L* and for *m*. A <u>good</u> value of *L* and *m* is a value such that "$E(\bar{X}_{Lm}(j))$ is sufficiently near $\mu$ to allow us to *treat* the $\bar{X}_{Lm}(j)$'s as being i.i.d. (independent and identically distributed), and unbiased for $\mu$ in the context of their use in a statistical inference problem, e.g., c.i. formation." (Kelton and Law, 1983). At the same time, the value of L cannot be too large, as this would mean excessively long computer time, and waste of resources. In other words, the problem addressed here is not a problem of optimisation, but of determining a set of values (L,m) such that the observations used to estimate $\mu$, may give a good estimate, and an acceptable confidence interval. "This way of thinking about the startup problem differs from that in Gafarian, Ancker, and Morisaku (1978), where the problem is defined as finding the minimal i* such that $E(X_i)$ is within a specified tolerance of $\mu$ for all $i \geq i^*$; their formulation requires that *individual* points be near $\mu$ in expectation, whereas our goal is to obtain *averages* of points which have expectation near $\mu$ ." (Kelton and Law, 1983)

### 4.3.2. KELTON AND LAW'S METHOD: DESCRIPTION

"If $E(X_i) \approx \mu$ for $i \geq q$ ($q$ unknown), then in econometric parlance, a *model* for $\bar{X}_i$, for $i \geq q$, is:

$$\bar{X}_i = \mu + \eta_i \qquad (1)$$

where the $\eta_i$'s are r.v.'s" (random variables) "with $E(\eta_i) = 0$. If we were to fit a straight regression line through adjacent $\bar{X}_i$'s over values for $i \geq q$, we would expect the fitted line to have a slope which would not be distinguishable from zero, upon performing a formal hypothesis test for zero slope. This is really a test for flatness of the TEF, (*transient expectation function* is the plot of $E(X_i)$ against i) so should indicate whether the TEF has stabilized, and this stability should only obtain at the level of $\mu$, in view of our assumption that the TEF is monotone. A serious difficulty, however, in fitting such a regression line and performing this test is that the $\bar{X}_i$'s are correlated, so that the $\eta_i$'s in Equation (1) are also correlated. This is contrary to the usual independence assumption made in classical regression so that we cannot simply apply ordinary least squares (OLS) to fit the desired line. Instead, we must resort to generalized least squares (GLS), which allows for autocorrelation in the disturbance terms of the regression model (See Johnston, 1972). A practical, general, and efficient GLS procedure was given by Amemiya (1973) which results in an unbiased and efficient (in the sense of minimum variance) slope estimator, and an asymptotic theory on which standard error estimates of this slope estimator can be obtained, which enables us to perform the desired zero-slope hypothesis test." (Kelton and Law, 1983).

Law and Kelton describe the method as follows:

"To state the procedure, we will need the following notation:

k = number of replications

$m_0$ = initial length of each of the k replications

$\Delta m$ = number of points added to each one of the k replications (if necessary).

$m^*$ = maximum replication length.

b = number of batches

$p^*$ = maximum initial deletion proportion.

$p_0$ = minimum initial deletion proportion.

$\beta$ = size of the test for zero slope.

f = maximum number of segments over which a fit is made, including initial fit.

The idea behind choosing *which* segments to use for curve fitting is to start near the *end* of the $\bar{X}_i$ series (with, for example, the last half of the data) for the initial fit, then move the segment *backward* toward the beginning of the data until it appears that the TEF is no longer flat, as evidenced by rejection of the null hypothesis of zero slope. If the line fitted initially to, for example, the last half of the data, has a slope estimate which is significantly different from zero, then *m* must be increased, i.e., each of the *k* replications must be extended, and we try again...

Before giving a detailed statement of the procedure, one other idea warrants discussion. Instead of fitting the regression lines to the $\bar{X}_i$'s themselves, we instead group the *m* $\bar{X}_i$'s into *b* "batches" to form *b* batch means, each being the average of *m/b* adjacent values of the $\bar{X}_i$ series; these batch means then form the points on which the regressions are done...

The initial line is fit to the last 100(1-$p^*$)% of the data, so uses the last (1-$p^*$)b batch means. Assuming that the initial zero slope test does not result in rejection, we begin consideration of earlier batch means by moving the interval over which the next line is fitted backward toward the beginning of the time series. To do this, the deleted proportion is reduced by an amount $\Delta p = (p^* - p_0/(f-1)$, so that the next line is fitted to batch means $(p^* - \Delta p)b$ through $(1 - \Delta p)b$, i.e., the right endpoint is also moved back. If this new line also has a slope which cannot be distinguished from zero, the next line is fitted to batch means $(p^* - 2\Delta p)b$ through $(1 - 2\Delta p)b$. As long as the zero slope tests do not indicate rejection, we keep diminishing the deletion proportion by $\Delta p$ each time until rejection occurs, or until the deleted proportion reaches $p_0$. Thus, we will do at most *f* fits in this way, and the interval of interest moves back by the constant amount of $(\Delta p)b$ batches for each fit.

Also, we assume that each value of *m*, the replication length, is divisible by *b*. Furthermore, it is understood that whenever an index is defined in a way in which it might be nonintegral, it is rounded to the nearest integer. Finally, the notation "$\leftarrow$" is to be read "is replaced by." Then the procedure is as follows:

STEP 1. Make *k* independent replications of length $m_0$ points each, average over the replications to obtain the single time series, $\bar{X}_1$, $\bar{X}_2$... $\bar{X}_m$ and let $m=m_0$.

STEP 2. Group the *m* points $\bar{X}_1$, $\bar{X}_2$ ... $\bar{X}_m$ into *b* batches of *m/b* adjacent points each, and compute the *b* batch means.

STEP 3. Fit a straight line through batch means p*b+1...b (using the Amemiya GLS procedure), and perform a test for zero slope at level ß.

a. If the test fails to reject the null hypothesis of zero slope, go to step 4.

b. If the test indicates rejection, then:

   i. If $m + \Delta m \leq m^*$ then $m \leftarrow m + \Delta m$, and go to step 2.

   ii. If $m + \Delta m > m^*$, print a warning that $m^*$ is too small, set $p = p^*$ and go to step 6.

STEP 4. Let $\Delta p = (p^*-p_0)/(f-1)$ and let $p=p^*-\Delta p$.

STEP 5. Fit a straight line through batch means pb+1, ... ,(p+1-p*)b and perform a test for zero slope at level ß.

a. If the test fails to reject, then

   i. If $p - \Delta p \geq p_0$ Then $p \leftarrow p-\Delta p$, go to step 5.

   ii. If $p - \Delta p<p_0$ then go to step 6.

b. If the test indicates rejection then $p \leftarrow p + \Delta p$ and go to step 6.

STEP 6. Let L = pm (to the nearest integer) and return L and m." (Kelton and Law, 1983).

The first problem with this method is the need to define values for some of the parameters, like for example $m_0$, $\Delta m$, b, p*, $p_0$, ß, f. Assigning values to parameters in simulation may sometimes be a problem, as shown in Chapter 5 for the batch means method, because the parameters may be model dependent or even worse, these values may depend on the unknown quantity being estimated by the simulation. The authors themselves recognise that their selection of parameters may not work well "in *every* case" (Kelton and Law, 1983).

A second problem is the lack of generality as this method was tested with "a total of 13 stochastic models with known $\mu$". (Kelton and Law, 1983). However, in typical simulation applications, the real value $\mu$ is not known. If it were known there would be no need to use simulation.

A third problem with this method is, as reported in a survey made by Kimbler and Knight (1985), that the method is rather complex: "In fact, had we not been able to obtain a written coding for the Amemiya GLS method our work would have been greatly exaggerated."

Kelton and Law (1983) give the following example of the procedure's operation, which is illustrated in Figure 4.2.

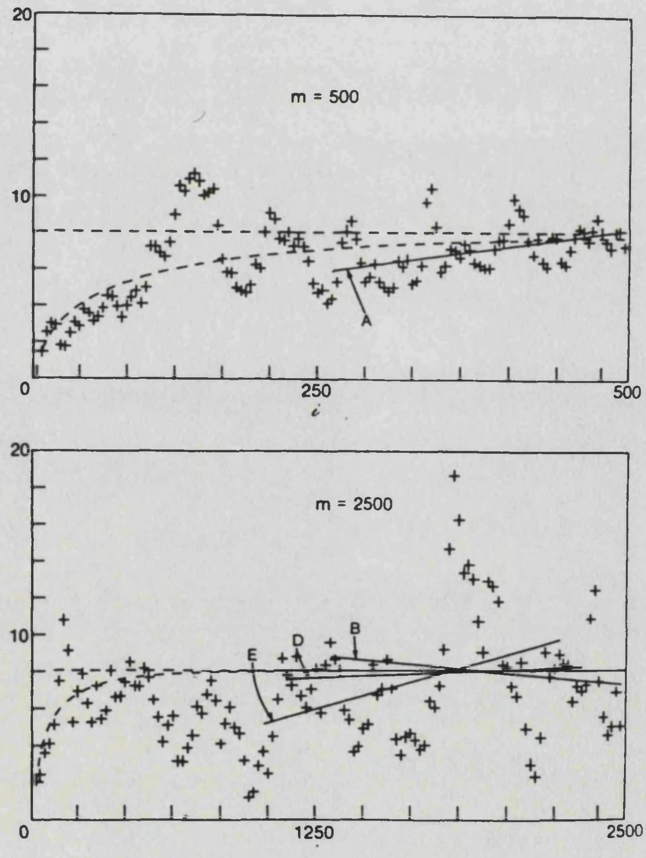



**Figure 4.2.** Law and Kelton's method: example for an M/M/1 queue with $\tau$ = 0.9.

"The process being simulated is again the delay in queue for the M/M/1 queue with $\tau = 0.9$ and the initial conditions are empty and idle. We implemented the procedure with the following parameters: $k = 5$, $m_0 = 500$, $\Delta m = 500$, $m^* = 3000$, $b = 125$, $p^* = 0.5$, $p_0 = 0.1$, $\beta = 0.5$ and $f = 11$. The top graph of Figure 4.2. pictures the initial situation, where $m = m_0 = 500$, so the batch means are averages of $m/b = 4$ adjacent $\bar{X}_i$ values each; these are plotted as the crosses, and the dashed lines give the level of $\mu$ and the exact TEF. The initially fitted line, labeled "A," is fitted to batch means 64 through 125, and the zero slope test indicated rejection at the $\beta = 0.5$ level. The procedure thus executes Step 3b(i), continuing each of the five replications for an additional $\Delta m = 500$ points, and extends the averaged time series to $\bar{X}_1 \ldots \bar{X}_{1000}$. Step 2 then forms 125 batch means of eight $\bar{X}_i$'s each and a new line is fit to these means of (larger) batches 64 through 125. In this example, the zero slope test again indicated rejection; we omit the plot in this case. The value of $m$ was increased in this way, until finally $m = 2500$ was reached, and Step 2 formed 125 batch means of 20 $\bar{X}_i$'s each; this is depicted in the lower graph of Figure 4.2. The line fitted to batch means 64 through 125 by Step 3 finally has slope which is not significantly different from zero; this is line "B." Step 3a thus sends us to Step 4, where the interval is moved back toward the beginning of the data. The next line fitted is not plotted, for clarity, but also resulted in not rejecting the null hypothesis of zero slope. The following line, labeled "D" also has an insignificantly nonzero slope. Moving back further, however, results in line "E" which leads to rejection of the zero slope hypothesis. Step 5b thus readjusts $p$ to indicate the beginning of the most recently fitted line which still appears to be flat, and the procedure returns with $L = 1050$ and $m = 2500$." (Kelton and Law, 1983)

This simple example shows that the method is not simple and easy to use and it confirms our previous discussion on the problems of the method. It requires complex programming and the setting of parameters that may be model dependent. Even more important from a practical point of view is the answer to the question: does it work only with simple models or can it be used for more complex models? And if it is used for complex models how long does the estimation of L and m take ? These are important questions that if not answered satisfactorily can make the procedure of little practical value.

### 4.3.3. WELCH'S METHOD

In this section we describe a second procedure that has been proposed for dealing with the initialisation bias problem and that according to Law (1983) "seems promising". Although this method is not as complex as Law and Kelton's method, it still requires some additional computing and the setting of some parameters that need to be determined by trial and error. Due to the complexity of some simulation models, this setting is not always easy as values tend to greatly differ from parameter to parameter and from model to model. The method is described by Law (1983) as follows:

"When the steady state average response $\upsilon$ exists, it is also given by $\upsilon = \mathrm{Lim}_{i \to \infty} E(Y_i)$. The goal of Welch's (1981,1983) procedure is to determine an index, say $L_0$, such that $E(Y_i) \approx \upsilon$ for $i > L_0$. Then $L_0$ is the number of observations that is to be deleted from the beginning of each simulation run. The value $L_0$ can be given two interpretations. First it might be considered to be a time index beyond which the process $Y_1, Y_2 ...$ is approximately covariance stationary. Also, the determination of $L_0$ facilitates obtaining an unbiased point estimate for $\upsilon$; in particular, the sample mean of the observations $Y_{L0+1}$, $Y_{L0+2}$ ... $Y_{L0+t}$ should be an approximately unbiased estimator for $\upsilon$.

In general it is impossible to determine $L_0$ from a single replication of the process because of its inherent variability. As a result, Welch's procedure suggests making $n$ independent replications of the simulation ($n \geq 5$) each of length $m$ observations. Let $Y_{ji}$ be the ith. observation from the *jth.* replication ($j=1, 2...n$; $i=1, 2...m$) and let

$$\overline{Y}_i = \sum_{j=1}^{n} \frac{Y_{ji}}{n} \qquad i=1,2...m \tag{4.3}$$

Observe that $E(\overline{Y}_i) = E(Y_i)$ and $Var(\overline{Y}_i) = Var(Y_i)/n$; thus, the process $\overline{Y}_1, \overline{Y}_2 ...$ has the same expectations (and correlation structure) as the original process, but is less variable.

To smooth out the high frequency oscillations in the averaged process (but leave the low frequency oscillations in which we are interested), we further define the *moving average* $\bar{Y}_i(w)$ by equation 4.4., where w, the "window" of the moving average, should satisfy $10 \le w \le [m/2]$.

$$\bar{Y}_i(w) = \begin{cases} \sum_{k=-(i-1)}^{i-1} \frac{\bar{Y}_{i+k}}{[2(i-1+1)]} & \textit{if } (i<w+1) \\ \sum_{k=-w}^{w} \frac{\bar{Y}_{i+k}}{[2w+1]} & \textit{if } w+1 \le i \le m-w' \end{cases} \tag{4.4}$$

Then $\bar{Y}_i(w)$ is plotted for i=1, 2...m-w and $L_0$ is chosen to be that value of *i* beyond which $\{Y_i(w)\}$ appear to have converged. <u>The values of *n, m, and w* need to be determined by trial and error.</u>

In Figure 4.3. we illustrate an application of Welch's procedure to the process $D_1$, $D_2$ ... for the M/M/1 queue with $\tau = 0.8$. The overall objective of the simulation study was to determine $d = \text{Lim}_{i\to\infty} E(D_i) = 3.2$ and here we chose $m = 470$, $n = 25$ and $w = 20$. (The vertical lines in Figure 4.3. show the 90% c.i's.) From the plot, we subjectively choose $L_0$ to be 150. One drawback of Welch's procedure is that it might require a large number of replications to make the plot of $\{\bar{Y}_i\}$ reasonably stable if the process $\{Y_i\}$ is highly variable." (Law, 1983).

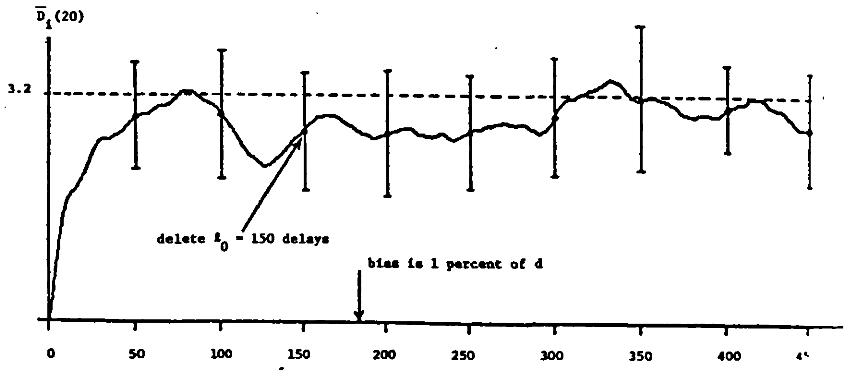



Figure 4.3. Welch's method applied to the M/M/1 queue with $\tau = 0.9$.

As with Law and Kelton's method, Welch's method requires the setting of some parameters. This step may not be obvious, and may be model dependent. And if they have to be determined by trial and error, the time spent in the estimation of the number of observations to be deleted may be too long, especially if we are simulating a rather complex system.

To summarise this section, from the computational point of view, both methods require some additional computations when compared to the method proposed later in this chapter. Although this may not be important for simple simulation models, the additional computation may be time consuming for complex systems. The main drawback of these methods, specially in Law and Kelton's, is their complexity. Implementing them will require extra effort from the simulation practitioner and will also require some extra computer time. It is important to notice that complex methods, should, if possible, be avoided in this stage of the simulation, because of the additional computer time that they require. Elimination of the initialisation bias problem is only a small part of the simulation itself; therefore, the computer time spent in dealing with it should be only a small fraction of the total computer time spent in the project (i.e., elimination of the influence of the initial conditions, running the simulation, and analysing the results).

Even though only two methods have been described here, the problem of the initialisation bias is extensively discussed in the literature. Procedures have been proposed that, using some of these tests to eliminate the initial bias, let the simulation practitioner calculate a confidence interval of pre-specified width (Heidelberg and Welch, 1983). Additional information on the problem and on different procedures can be found in Schruben and Singh (1983), Schruben (1982), Glynn and Iglehart, (1987). For procedures proposed before 1978, see Gafarian et al, (1978) and Wilson and Pritsker, (1978a).

## 4.4. A NEW METHOD FOR THE ELIMINATION OF THE INITIALISATION BIAS PROBLEM

Three main problems were identified with the methods described in

Section 4.3: insufficient testing for some complex models with no analytical answer, complex methods that may require a large computer time just to deal with a small part of the total simulation problem and the need to define some parameters whose values may be model dependent. These problems arise from the fact that in typical simulations there are several entities in the model and they interact with each other; in simple models with known analytical answer, this interaction does not exist, or is minimal. In the method proposed here there are no parameters to be defined and in this way the possible dependency of it on the simulation model is eliminated. Likewise the method is tested for some typical simulation models (i.e., no known answer), as well as for some common models like the M/M/1 queue, the M/M/4 queue and a system of queues in tandem. A last advantage of the new method is that it is easily implemented as is shown in section 4.4.3.

Section 4.4.1. explains the basic idea behind the new method. Section 4.4.2. explains a method proposed by Gordon (1969) based on the variation of the STANDARD DEVIATION and which, being similar to the one proposed in this chapter, will be used to compare our results with those of an existing method. In this section we also describe one of the first methods proposed in the literature by Conway (1963) for the estimation of the number of initial observations to be deleted. Section 4.4.3. develops an algorithm for the method proposed in this chapter. Section 4.4.4. compares and points out the differences between Gordon's method and ours.

### 4.4.1. PROPOSED NEW METHOD : ITS BASIS.

The discussion of the previous sections, especially of Section 4.3., shows the need for a method that is simple and easy to implement for dealing with the initialisation bias problem. At the same time the new method should not be time consuming. Following the empirical approach used in this research we studied the behaviour of several simulation models for short simulation run lengths. If $X_1$, $X_2$...$X_k$ are the mean estimates obtained from replication i, i=1, 2...k for a particular simulation run length, *t*, the mean estimate $(\bar{X}_t)$ can be calculated as the average of the $X_i$'s. We observed that the STANDARD DEVIATION of the mean estimate tends to increase for short simulation run lengths (i.e., for small values of $t$); as the simulation run length increases (i.e., as the value of $t$ becomes larger) it will reach a maximum value and for longer simulation run lengths the standard deviation decreases. If the initial state is close to the steady state we can expect an increase in the value of the standard deviation (as the variation is minimum at the beginning of the simulation), followed very soon by a decrease in its value, as we would expect from basic statistics. But when the system is started "idle" and "empty" the influence of the initial conditions is strong and the standard deviation will take longer to start decreasing. Following this trend of thought we decided to use the point in simulated time for which the standard deviation reaches a maximum as the run-in-period. Experiments using this approach show, as will be explained later, that using the value of the run-in-period thus determined the parameter of interest will reach the steady state for shorter simulation run lengths and that the difference between the estimate and the real value is smaller than when other run-in-periods are used.

It is important to notice that the idea behind the new method proposed in this chapter originated from an empirical observation of the behaviour of different systems. Some heuristic methods proposed in the literature are based on theoretical considerations, like the one proposed by Gordon (1969). In this method the identification of a truncation point is based, like ours, on the analysis of the variation of the standard deviation of the mean estimate as the simulation run length increases. Because of its similarity to the one we propose we describe it below, and use it to compare the performance of our method to an existing one.

### 4.4.2. GORDON'S AND CONWAY'S RULES

In Gordon's rule we make $k$ replications; each replication has a sample size $n$, i.e., $n$ observations ($x_i$, i=1, 2..n) are recorded in each replication. The value $\overline{y}_j$ corresponds to the average over $n$ observations, in replication $j$. If $\overline{X}_n$

is the average of the values $\overline{y}_j$, j = 1, 2 ...k, we choose the truncation point L=n, for which Var[$\overline{X}_n$] begins to fall off as 1/n.

This can be expressed in a mathematical form as follows:

$$\overline{y}_j = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{4.5}$$

and the sample mean estimate $\overline{X}_n$ is given by Eq. 4.6.

$$\overline{X}_n = \frac{1}{k}\sum_{p=1}^{k} \overline{y}_p \tag{4.6}$$

Similarly, using classical statistics formulas, the variance of this mean estimate can be estimated by $s^2$ as shown in Eq. 4.7.

$$s^2 = \frac{1}{k-1}\sum_{p=1}^{k} (\overline{y}_p - \overline{X}_n)^2 \tag{4.7}$$

From these equations we see that the variance is inversely proportional to (k-1) and according to Gordon the variance will fall as 1/n once the initial bias has disappeared. The reason for this assumption is that if only one replication is made, the mean queuing time, for example, would be estimated by accumulating the waiting time of *n* successive entities and dividing by *n*. The variance of the mean estimate is inversely proportional to *n* and the standard deviation to $n^{-1/2}$. In practice, due to the variability in the value of the estimate if only one replication is used, Gordon proposes to make *k* replications, but the variance is still a function of (1/n). To identify the cut-off point, i.e., when the variance falls by 1/n, or equivalently the standard deviation falls by $1/n^{1/2}$, a graph of Log(s(n)) vs Log(n) is made, and the number of initial observations to be deleted is given by the value of *n* at which the graph becomes approximately linear with slope -½. The method is illustrated in Figure 4.4. where Gordon's rule has been applied to the M/M/1 queue with traffic intensity $\tau$ = 10/15 = 0.666. The slope of the graph becomes -½ for approximately Log(n)=2, which corresponds to n=100.

There are four main problems with this method. The first one is that from its evaluation made by Gafarian et al (1978) (see as well Wilson and Pritsker, 1978b) it appears that the truncation point tends to be overestimated although these evaluations have been made only for simple models. The second problem is that Gordon does not support his procedure with empirical results, except one or two for the M/M/1 queue. A third problem is that in practice, the fall of the standard deviation will rarely have a constant slope close to ½. The values of this slope tend to oscillate in a rather large range and therefore identifying a particular point at which the slope becomes -½ is very difficult. Appendix E uses Gordon's method for the estimation of the run-in-period of the simulation models used to test the behaviour of the procedure proposed in this chapter for dealing with the initialisation bias problem, and it shows how very rarely the fall has a constant slope of -½.

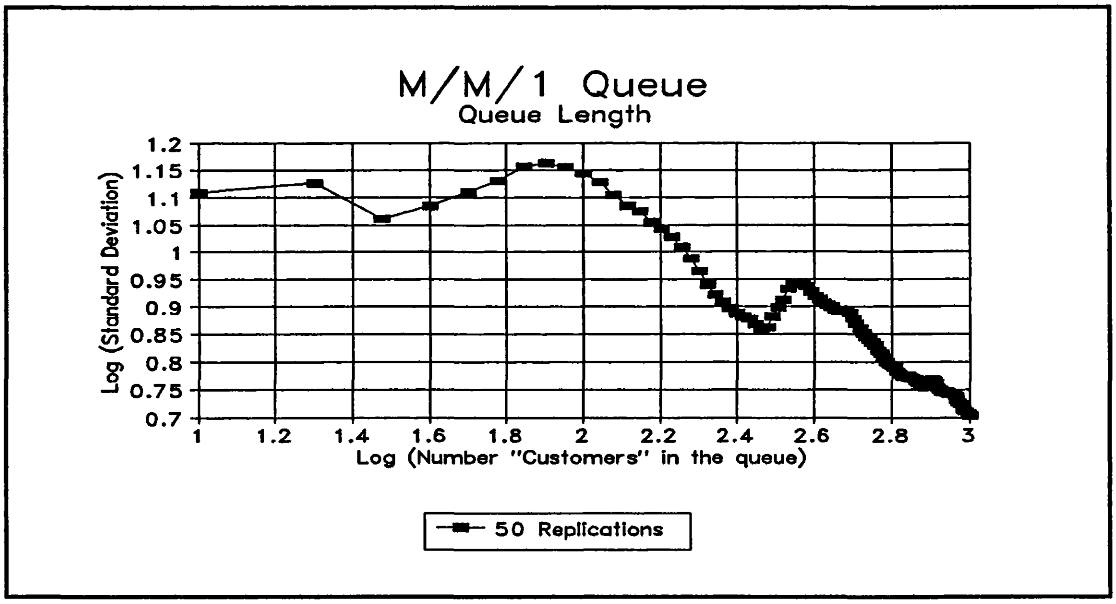



**Figure 4.4.** Logarithm of the standard deviation of the mean queuing time as a function of the logarithm of the number of observation in the M/M/1 queue.

However, the most significant of all the problems with Gordon's method is that, like the mean estimates, the standard deviation values are affected by the number of replications. And not only the values, but also the rate of change in the values, will be different depending on the number of replications, as is shown in Figure 4.5. This figure shows the values of the standard deviation obtained as a function of the number of replications and of the number of "customers" that are waiting to be served in the WASHQ queue of the LAUNDERETTE model (See Figure A.5, Appendix A). As can be seen from this figure, the slope of the graph depends on the number of replications. Although the results are not given in this thesis, as we increase the number of replications, its influence on the values and the slope of the graph of the standard deviation is less. This conclusion is similar to the one discussed in Chapter 3, concerning the mean estimates. As is shown in Chapter 3, the number of replications required for this influence to be negligible depends on the model, and on the parameter. Therefore, Gordon's method will not work well for different types of models. But, as can be seen from the graph, the simulated time for which the standard deviation reaches its maximum is not greatly influenced by the number of replications.

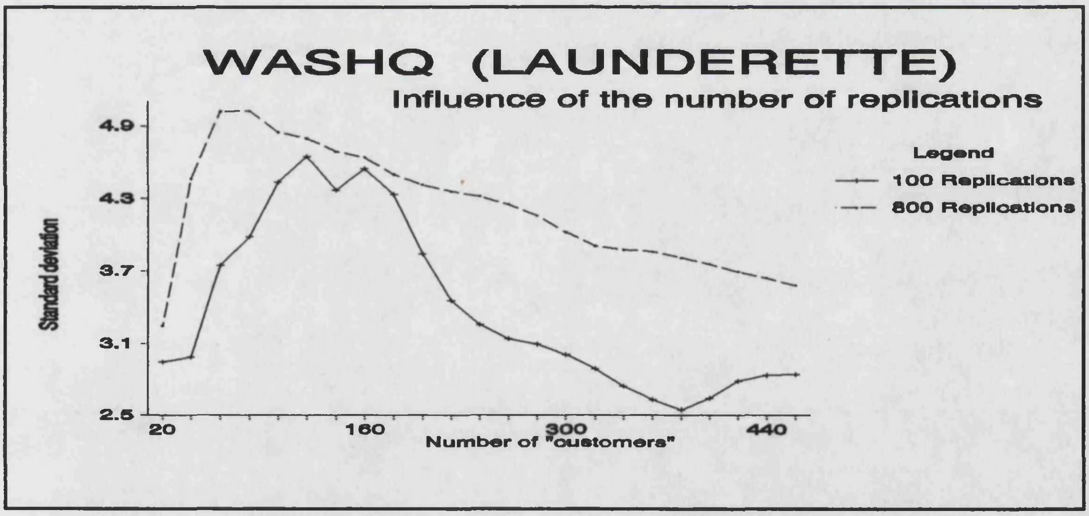



**Figure 4.5.** STANDARD DEVIATION of the WASHQ mean queuing time estimates, as a function of the number of replications, and of the number of "customers" being served.

Another method that will be used in the following sections is Conway's method. This method was proposed in 1963 and is described by its author as follows:

"I usually truncate a series of measurements until the first of the series is neither the maximum nor the minimum of the remaining set. I do not do this for every run but rather decide on a stabilization period by examining a few pilot runs and thereafter delete this same period of each run." (Conway, 1963).

This method is better explained by Gafarian et al (1978) as follows:

"This rule specifies a priori the number of exploratory replications and the number of observationns per exploratory run, denoted by *e* and *L* respectively. Then if $\{x_{i1}, x_{i2}, ...x_{iL}\}$ is the set of observations on the ith. exploratory run, one computes

$$x_{ik}^{+} = \max \{x_{ik}, x_{ik+1}, ...x_{iL}\}$$

and

$$x_{ik}^{-} = \min \{x_{ik}, x_{ik+1}, ...x_{iL}\}$$

for k = 1, 2 ...L and determines $t_i$ such that

$$x_{iti}^{-} < x_{iti} < x_{iti}^{+}$$

occurs for the first time. Then the estimate of t* (number of initial observations to be deleted in each replication) is given by

$$t^* = \max \{t_1, t_2, ... t_L\}$$ " (Gafarian et al, 1978)

This method is easier to use than Gordon's method, but it seems from the tests performed by Gafarian et al, result that has been confirmed in this research, that it greatly underestimates the truncation point.

### 4.4.3. USE OF THE STANDARD DEVIATION TO ESTIMATE A "GOOD" RUN-IN-PERIOD : AN ALGORITHM

As discussed in Section 4.4.1. the estimation of the run-in-period can be done by studying the change in the value of the variance of $\bar{X}_t$, as the simulation run length increases. $\bar{X}_t$ is the average of the mean estimates obtained from *k* different replications when the simulation run length is *t* units of time.

If the method is going to be of practical use, considering that we are dealing with discrete simulations, we cannot estimate the standard deviation for each different value of simulated time. In practice we only obtain estimates every $T_1$ units of time; the value of $T_1$ cannot be too large as the estimate of the run-in-period would not be accurate, but it cannot be too small as this would increase the computer time required to obtain this estimate, time that we want to keep small. Rules for selecting the value $T_1$ are given in the algorithm below, and they are based on the empirical results obtained while developing the new method.

Due to the randomness inherent to simulation there may be small increases in the value of the standard deviation after it has reached its maximum. We found that this problem is overcome and the analysis is made easier if we study the change in the values of the standard deviation (Y-axis) as a function of the simulation run length (X-axis) in a LINE GRAPH.

We should notice that the value of the standard deviation is influenced not only by the initial conditions, but also by the number of replications *k* as discussed in section 4.4.2. But we found from empirical results that although the rate of change in the values of the standard deviation are affected by the number of replications, the simulated time for which the standard deviation reaches its maximum value does not change too much with the value of *k*. Therefore, as the computer time used for the calculation of the run-in-period should be kept as small as possible, we suggest not to take more than 20 to 30 replications.

We can formulate the above discussion in a formal way in the following **algorithm.** It explains the steps to be followed when the new method is to be used for the estimation of a run-in-period as a method for dealing with the initialisation bias problem.

***ALGORITHM***

1. Choose an initial period of time $T_0$, and divide it into N intervals. Therefore, each interval has a length of $T_1 = T_0/N$ units of time.

2. Make *k* replications and obtain an estimate of the mean for each one

of the N intervals. Call $X_{ij}$ the mean estimate in the ith. replication (i=1, 2...k), for a simulated time $j*T_0/N$ (j=1,2..N). This means that $X_{ij}$ is the *cumulative sample mean* for a given simulation run length $j*T_0/N$, as it takes into account all the previous observations.

3. Estimate the mean for each one of the N intervals of time as the average of $X_{ij}$ over the *k* replications, applying Eq. 4.8.

$$\overline{X}_j = \frac{1}{k}\sum_{i=1}^{k} X_{ij} \tag{4.8}$$

4. Estimate the standard deviation for each one of the mean estimates obtained in step 3 by applying Eq. 4.9.

$$s_j = \sqrt{\frac{1}{k-1}\sum_{i=1}^{k}(X_{ij}-\overline{X}_j)^2} \tag{4.9}$$

5. The run-in-period is estimated by the value of $j*T_0/N$ for which the STANDARD DEVIATION reaches a maximum value. We will call this value of simulated time t*.

NOTES ON THE ALGORITHM:

1. The method proposed here for dealing with the initialisation bias problem is an "off-line" method. This means that it should be used previous to running the steady state simulation. In other words, if we suspect that the influence of the initial conditions should be eliminated (see discussion in Chapter 2), before running the steady state simulation, we use the algorithm as proposed above. We estimate a time t*, also called *run-in-period,* and then we run the steady state simulation. In practice, although observations are recorded from the beginning of the simulation we discard those recorded during the run-in-period; we only use for the estimation of the parameter of interest those observations recorded for simulated times t>t*. It should be noted that although the usual approach to deal with the problem of the initialisation bias is to delete a fixed number of observations L in each

replication, in the method proposed in this chapter we estimate a fixed period of time t*. The number of observations recorded (and then discarded) during t* will be a random variable and will differ from replication to replication.

2. In order to estimate the run-in-period we need to determine the value of $j^*T_0/N$ for which the **standard deviation** reaches a maximum value. Although this analysis can be done by looking at the numerical values obtained from the algorithm, a graphical analysis is easier. In this case we simply obtain a **line** graph of $s_j$ on the Y-axis as a function of the simulation run length, $j^*T_0/N$ on the X-axis. The maximum value of $s_j$ is easier to determine in a **line** graph because, as we are dealing with stochastic systems, there may be some small increases in the value of the standard deviation for values of $j^*T_0/N$ greater than t*; however, after this slight increase, the decrease in its value will continue. While a visual observation of the values may not show clearly that this is just a slight increase, the graphical method permits an easier determination of the point where the value of the standard deviation reaches its maximum and then starts to decrease.

3. The use of a spreadsheet to draw the graph is not absolutely necessary. Using a high-level language, like PASCAL for example, it is easy to program and draw the type of graph that is required for this procedure.

4. If for the selected value $T_0$ the standard deviation has not reached a maximum, and if a previous analysis (see Chapter 2) has been carried out to rule out the possibility of the system, or at least of that particular parameter, to be unstable, then the maximum value should be increased say to $2^*T_0$, while keeping the length of each time interval constant. Usually the number N of intervals required for the standard deviation to reach its maximum value is directly related to the value of the traffic intensity $\tau = \lambda/(s\mu)$. The larger this value, the more the number of intervals N that are required. However, as is explained in Section 4.5.5. in more detail, there may be some problems associated with queues with large values of $\tau$.

Similarly, when the initial conditions are similar to the steady state conditions the influence of the transient may be very small and in this case the graph of the standard deviation as a function of the simulation run length does not have a maximum value. In these cases, the practitioner should perform a quick test as described in Section 2.5.2. to determine if a run-in-period is necessary; if it is necessary then the run-in-period is chosen to be $T_1$.

5. The length of the intervals $T_0/N$ may be chosen so that its value is slightly larger than the maximum activity execution time which can be defined as follows:

"*Maximum Activity Execution Time*": As we know, the different activities of the simulation model take some time to be executed. In some cases this time is deterministic, but in most cases it follows a given probability distribution, with a specified mean. The Maximum Activity Execution Time is the largest of these means.

The initial number of intervals may be chosen to be 30, even though this value might have to be increased depending on the influence of the initial conditions (See NOTE 4). These values are based *on empirical results*. The advantage of this method is that this choice of parameters is not critical; for example, in the simulation of the PUB, the length of the intervals is chosen to be four times the maximum activity execution time and the results are still quite reasonable. This point will be explained more in detail as we go through the different examples used to test the new method.

6. As discussed above, the number of replications $k$ does not need to be large. On the other hand, due to the variability of the mean estimates, especially for short simulation run lengths, it cannot be too small. From empirical results we found that taking 20 or 30 replications gives good estimates for the run-in-period as is shown in the examples of this chapter and of Appendix F. There are two possible cases when the simulated time for which the standard deviation reaches a maximum is influenced by the number of replications. These two cases are discussed in Section 4.4.5.

7. It is important to emphasise the fact that, as with most methods proposed in the literature to deal with this problem of the influence of the initial conditions, we do not attempt to obtain an "optimal" run-in-period, but a "good" one where the influence of the initial conditions is negligible.

### 4.4.4. COMPARISON OF OUR METHOD AND GORDON'S METHOD FOR THE ESTIMATION OF A RUN-IN-PERIOD.

The method proposed in this chapter does not have any of the problems discussed in section 4.4.2. concerning the use of Gordon's method; what is more important is that from empirical results it can be shown that it works quite well for different types of simple and complex simulation models. This means that the number of observations that are deleted is not as large as it would be if Gordon's method were used. This has an important implication: the total simulation run length that is required to obtain a mean estimate close enough to the real but unknown value $\mu$ is shortened.

The first difference of the new method as compared to Gordon's method is related to the type of simulation software used in this research. As has been explained before, the attempt here is not to determine the number L of observations to be deleted, but a period of time (run-in-period) such that any observations made before it are discarded. Due to the use of different random number seeds in different replications, the number of observations recorded during the run-in-period is a random variable that changes from replication to replication (or vice versa, different simulation run lengths are required to record the same number of observations in different replications).

The second difference is that in Gordon's method the truncation point is given by the point of simulated time where the variance starts falling with a slope of 1/n, while in the method proposed in this chapter the truncation point is given by the point in simulated time for which the standard deviation reaches its maximum value. But the rate of fall at the beginning of the decrease in the value of the variance may be different from 1/n and consequently, in Gordon's method the truncation point may happen after a longer period of time than in the method proposed here. This point is confirmed in the empirical results where in all the examples considered in this chapter the truncation point in the method herein proposed will occur earlier in simulated time than in Gordon's method; what is more important is that the parameter will reach the steady state for a shorter simulation run length if the run-in-period estimated with our method is used than if Gordon's run-in-period is used.

### 4.4.5. POSSIBLE PROBLEMS WITH THE PROPOSED METHOD

Basically the main problem with the proposed method is if the simulated time for which the standard deviation of the parameter is influenced by the number of replications as occurs with Gordon's method. Under normal circumstances this should not occur. However, through thorough experimentation we have identified two possible cases where this may happen:

1. Sometimes, a certain combination of random number seeds may cause extremely large or small values in the simulation output data. By observing the values $X_i$, obtained in replication *i*, it is possible to identify the particular combination of random number seeds that will produce the outliers. By avoiding this combination the problem disappears. This is shown graphically in Figure 4.6. This figure shows the graph of the standard deviation of the mean queuing time parameter of the WAIT queue in the PUB model (Figure A.4., Appendix A) as a function of the simulation run length and of the number of replications: 20 or 40. In the case of 40 replications we observe how when the set of random number seeds R.N.1. is used, there is a sudden increase in the value of the standard deviation corresponding to a simulation run length of 1240. Looking at the values $X_1$, $X_2$, ... $X_{40}$ obtained from the different replications, we observed that $X_{24} = 5.67$, while the other values of $X_i$, $i \neq 24$, are all in the range [0.69 , 1.56]. We then replaced the combination of random number seeds that was producing this outlier, and obtained the mean estimates corresponding to the set of values R.N.2. (this set of values is therefore identical to the set of random number seeds R.N.1. except in one combination). In this case, there is no sudden increase in the value of the standard deviation. Therefore, a successful use of the method proposed in this chapter requires the identification of random number seeds (one or maximum two) that may produce outliers in the simulation output. When the random number seeds that produce, in this particular case the

outlier $X_{24}$, are used under different input conditions even in the same model the outlier will not necessarily occur. Even more, it may be possible that the problem of the outlier will occur only for some simulation run lengths and not for every run length. Identification of this phenomenon is only possible with an analysis of the individual values obtained from the simulation output and that will be used to obtain the mean estimate, $\bar{X}$.

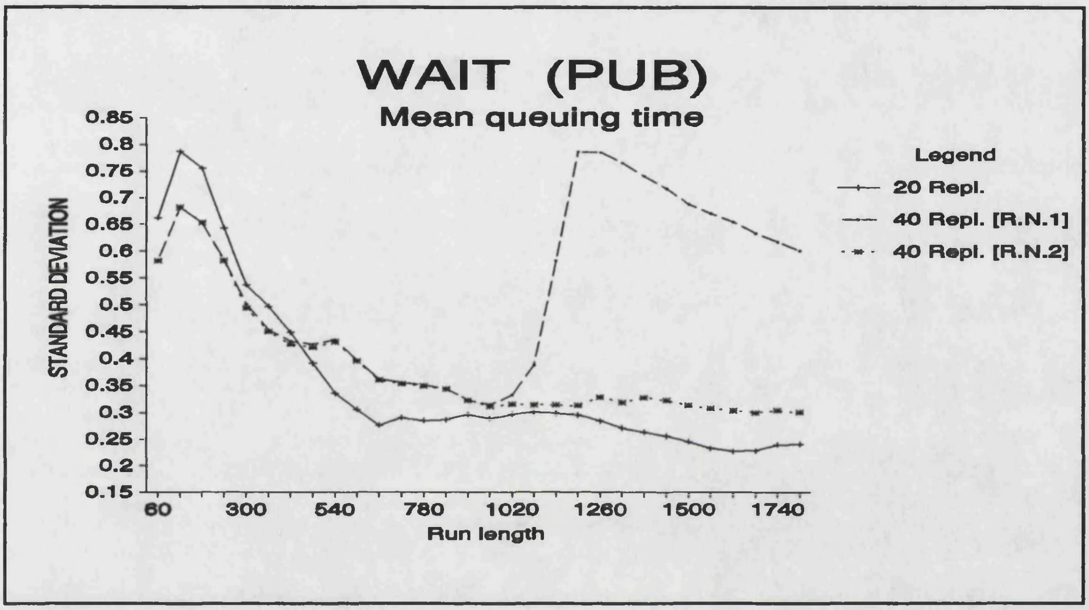



**Figure 4.6.** STANDARD DEVIATION of the WAIT mean queuing time estimates as a function of the simulation run length and of the number of replications.

2. A second reason for which the simulated time for which the standard deviation reaches its maximum may be influenced by the number of replications is related to the value of the traffic intensity. When the traffic intensity $\tau$ takes large values there are different maximums in the graph of the standard deviation as a function of the simulation run length if the number of replications used is small. As this number increased, the graph tends to show only one maximum value for the standard deviation. Basically the problem is due, as discussed in detail in Appendix F, to the large values of the standard deviation as compared to the values of the mean estimates. In cases like this,

where there are more than one maximum in the values of the standard deviation as a function of the run length it is necessary to increase the number of replications. This may increase in some cases the time required for the estimation of the run-in-period, but even with this problem, our method compares favourably to other methods, from the point of view of simplicity.

In other cases, the maximum value may change slightly depending on the number of replications. However, as has been emphasised in this thesis, the objective is not to obtain the **optimal** but a **good** run-in-period.

## 4.5. EMPIRICAL RESULTS.

In order to evaluate the algorithm proposed in Section 4.4.3. we applied it to different simulation models, and the results obtained are given in Section 4.5.2. (LAUNDERETTE) and 4.5.4. (M/M/1 queue). Additional results are given in Appendix F. For the evaluation of the performance of the proposed method, we define in Section 4.5.1. three measures of performance of a run-in-period.

### 4.5.1. MEASURES OF PERFORMANCE

In this section we define some measures of performance that may be used for the evaluation of the method proposed here (or any other method) for dealing with the initialisation bias problem. To test how well a run-in-period deals with this problem we need to know the real, but generally unknown, steady state value $\mu$. As has been explained in Chapter 2, we dealt with this problem in an empirical way. In Appendix C we give approximate values for $\mu$, for different queues and simulation models. From a practical point of view, we will consider a parameter to be in the steady state when all the mean estimates fall within $\epsilon = 2.5\%$ of this value of $\mu$.

With this convention in mind, we can define what a "good" run-in-period is. In practice the following three conditions should be met for a run-in-period to deal successfully with the initialisation bias problem:

1. When a run-in-period is used, the mean estimates obtained for short simulation run lengths (but obviously longer than the run-in-period) should be closer to the steady state value $\mu$ as compared with the mean estimates obtained when no run-in-period is used.

2. We are considering that the parameter has reached the steady state when the mean estimates fall within $\epsilon$ of the real steady state value. When the estimated run-in-period is used, the simulation run length required for the mean estimates to fall within this value $\epsilon$ should be shorter than the simulation run length required when no run-in-period or when shorter run-in-periods are used. At the same time, the use of a run-in-period longer than the estimated one will not shorten the simulation run length required for the parameter to reach the steady state.

3. In the case that for different run-in-periods the parameter reaches the steady state for approximately the same simulation run length, the run-in-period giving estimates closer in absolute value to the steady state value performs better.

### 4.5.2. ANALYSIS OF THE RESULTS FOR THE LAUNDERETTE AND THE MILITARY MODELS

In this section we apply the procedure developed in Section 4.4. to the LAUNDERETTE model, and we also discuss the MILITARY model. Other results for simulation models for which no analytical answer can be obtained are given in Appendix F. We discuss now the different results obtained for some of the queues of the LAUNDERETTE model.

#### 1. THE LAUNDERETTE MODEL: ANALYSIS

To illustrate the use of the method proposed in Section 4.4.3. and to evaluate its performance we obtained the run-in-period for the mean queuing time and the mean queue length of the DRYQ queue (See Figure A.5., Appendix A). In Appendix F we give similar results for other queues of this model. We will divide this study into two parts: the estimation of the run-in-period and the evaluation of its performance according to the measures of performance discussed in Section 4.5.1.

RUN-IN-PERIOD ESTIMATION FOR THE DRYQ QUEUE

We will obtain estimates of the standard deviation every $T_1$ units of time (minutes in this example). This value can be chosen to be slightly larger than the maximum execution time (Note 5, Section 4.4.3). Table 4.1. gives the probability distributions used to model the time taken by the different activities of this model to be executed (execution time). From this table the maximum activity execution time, considering only the mean value of the probability distribution, takes 40 minutes. We have then chosen $T_1 = 60$ minutes.

| **Activity** | **Execution time** (Probability Distribution) |
|---|---|
| ARRIVE | NEGATIVE EXPONENTIAL; MEAN : 8; |
| LOADW | 40; |
| UNLOADW | UNIFORM, between 1 and 5; |
| TRANSPORT | UNIFORM between 3 and 5; |
| LOADD | 4 |
| DRY | NORMAL; MEAN : 10; Standard Deviation : 3; |

**Table 4.1.** Execution time of the different activities of the LAUNDERETTE model.

Making 20 replications we obtained mean estimates and their corresponding sample standard deviation for the mean queuing time and the mean queue length of the DRYQ queue. Using a spreadsheet the values of the standard deviation as a function of the simulation run length were graphed, and the maximum values were recorded for both the queuing time and the queue length of the DRY queue.

Although the value of the simulated time for which the standard deviation reaches its maximum value can be found from the numerical values obtained from the simulation output, it is easier to identify it if we use a line graph as explained in Section 4.4.3. Figure 4.7. shows the graph of the standard deviation for the estimate of the mean queuing time of the DRY queue as a function of the simulation run length.

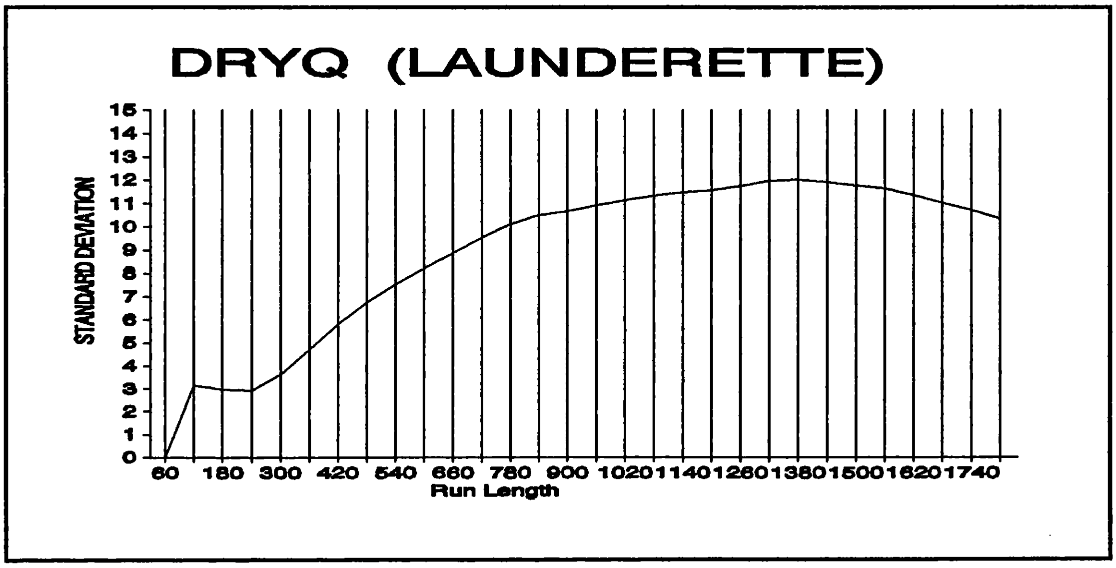



Figure 4.7. STANDARD DEVIATION of the Mean DRYQ Queuing time estimates, as a function of the simulation run length.

From this graph, the maximum value of the standard deviation occurs for a run length of 1380. Using a similar graph for the estimate of the standard deviation of the mean queue length parameter of the DRY queue we identify a run-in-period of 1380 for this case. That the run-in-period for the mean queuing time and the mean queue length parameters is the same for the DRYQ queue is a coincidence. In general, they might be different as, depending on the steady state mean values, some parameters will require longer run-in-periods to eliminate the influence of the initial conditions.

NOTE: Although the **standard deviation** has been graphed as a continuous variable, it is important to remember that it is a function of discrete values of the simulation run length, specifically it is a function of ($j*T_0/N$), where j=1, 2...

## RUN-IN-PERIOD PERFORMANCE: EVALUATION

To check how well the estimated run-in-periods for the mean queuing time and the mean queue length parameters perform in terms of the elimination or reduction of the influence of the initial conditions, we need to study the change in the mean queuing time and the mean queue length estimates as the simulation run length increases.

In our method we delete the number of observations that are recorded during the run-in-period. In Gordon's method we delete a fixed number of observations L in each replication. However, it is useful to be consistent and use a single measure for the comparison of both methods; in other words, we either obtain the mean estimates corresponding to number of observations in each queue of interest or we obtain the mean estimates for periods of time. At the same time, although the run length required to collect L observations is a random variable that takes different values in different replications the values do not differ too much from replication to replication. For these reasons we have approximated in Appendix E the run-in-period corresponding to the deletion of L observations using Gordon's method, where the value of L is estimated by the number "L" for which the graph of Log(s(L)) vs. Log(L) falls with a slope of -½. Using this approximation we obtained a run-in-period of 3780 corresponding to the application of Gordon's method to the mean queuing time and the mean queue length parameters of the DRYQ queue.

A similar approximation has been made for the run-in-period estimated using Conway's method. Using the method described in Section 4.2.2. we obtained that the number of initial observations to be deleted is 38, which corresponds to an approximate simulation run length of 300 minutes.

Following the empirical approach, we will obtain mean estimates for different run-in-periods, including the one estimated with the new method, as a function of the simulation run length. In this way, by using run-in-periods longer and shorter than the one estimated with the new method we can compare the performance of different run-in-periods. Hopefully, as discussed in Section 4.5.1. if different run-in-periods require approximately the same simulation run length for the parameter to reach the steady state, the estimates obtained with the run-in-period estimated with the method proposed in this chapter will be (usually) closer to the real steady state value.

In order to use the measures of performance defined in Section 4.5.1. we need the real steady state mean queuing time and mean queue length values ($\mu$); these were estimated in Appendix C; the parameters will be considered to be in the steady state when they fall within $\epsilon$ = 2.5% of $\mu$. The values of $\mu$ determined for the mean queuing time and the mean queue length of the DRYQ in Appendix C, as well as the range of values that fall within $\epsilon$ are the following:

| **Parameter** | **Steady state ($\mu$)** | **Range** |
|---|---|---|
| Queuing time | 17.670 | [17.228 , 18.112] |
| Queue length | 2.210 | [2.150 , 2.260] |

The system was simulated for a simulation run length long enough to show the approximate steady state value for the mean queuing time and the mean queue length of the DRYQ queue. We used the method proposed in Chapter 3 for the estimation of the number of replications required for the parameters to reach the steady state for the shortest possible simulation run length.

Tables 4.2a (mean queuing time) and 4.2b. (mean queue length) summarise the information recorded from the simulation output for the mean queuing time and the mean queue length estimates for different run-in-periods (Run-In) and for different simulation run lengths (3000, 5000, 7000...), including the case when no run-in-period (no deletion of observations) is used.

The column corresponding to the run-in-period estimated making use of the new method is indicated in Tables 4.2a and 4.2b. with "**". In these tables we have underlined the mean estimates for which the parameter can be considered to be in the steady state. This means that for the simulation run length corresponding to the underlined values, as well as for longer run lengths, the mean estimates all fall within 2.5% of the steady state value.

From the underlined values in Tables 4.2a. and 4.2b we notice that using the run-in-period estimated with the new method the mean queuing time and the mean queue length estimates fall in the above ranges for simulation run lengths as short as 6000 (queuing time) and 4000 (queue length); when no observations are deleted longer simulation run lengths (12000 or more) are required. Although it is true that for the mean queue length shorter run-in-periods could be used for the parameter to reach the steady state in approximately the same simulated time, the difference in absolute value between the mean estimates and the steady state value $\mu$ is in most cases smaller when the run-in-period estimated with the new method is used.

When the run-in-period estimated with Gordon's method (3780) is used we obtain the following mean estimates and corresponding standard deviation:

| | Queuing time estimates | | Queue length estimates | |
|---|---|---|---|---|
| Run Length | Mean | Std. Dev. | Mean | Std. Dev. |
| 5000 | 17.133 | 11.943 | 2.210 | 1.701 |
| 5500 | 17.176 | 11.187 | 2.210 | 1.587 |
| 6000 | 17.274 | 10.357 | 2.211 | 1.447 |

From this table we notice that the steady state for the mean queuing time parameter is not reached for a shorter simulation run length, as a simulation run length of at least 6000 minutes is still required. But the standard deviation of the mean estimate is increased as compared to the same estimate when the run-in-period estimated with the method proposed in this chapter is used; these values, that are not given in Tables 4.2a and 4.2b, are the following, for a simulation run length of 6000: 7.567 for the mean queuing time estimate and 1.037 for the mean queue length estimate.

With respect to the run-in-period estimated with Conway's method we notice that although the steady state seems to be reached for a shorter simulation run length when our run-in-period is used, but the difference is not too large.

| DRYQ mean queuing time estimates | | | | | |
|---|---|---|---|---|---|
| Run Length | Run-In 0 | Run-In 300 | Run-In 660 | Run-In 1380 | Run-In 1500 |
| | | | | ** | |
| 3000 | 15.594 | 16.300 | 16.662 | 16.592 | 16.426 |
| 4000 | 16.168 | 16.723 | 17.019 | 17.030 | 16.948 |
| 4500 | 16.291 | 16.854 | 17.082 | 17.107 | 17.092 |
| 5000 | 16.462 | 16.915 | 17.159 | 17.192 | 17.143 |
| 6000 | 16.655 | 17.036 | 17.241 | 17.280 | 17.246 |
| 7000 | 16.843 | 17.173 | 17.352 | 17.396 | 17.372 |
| 8000 | 16.939 | 17.229 | 17.386 | 17.425 | 17.405 |
| 9000 | 17.009 | 17.267 | 17.406 | 17.443 | 17.426 |
| 10000 | 17.060 | 17.294 | 17.419 | 17.453 | 17.438 |
| 11000 | 17.124 | 17.338 | 17.452 | 17.486 | 17.473 |
| 12000 | 17.224 | 17.421 | 17.527 | 17.563 | 17.553 |
| 13000 | 17.351 | 17.535 | 17.636 | 17.675 | 17.668 |
| 14000 | 17.417 | 17.589 | 17.684 | 17.723 | 17.717 |
| 15000 | 17.367 | 17.526 | 17.614 | 17.647 | 17.641 |

**Table 4.2a.** Mean DRYQ queuing time estimates as a function of the simulation run length and of the run-in-period.

| DRYQ mean queue length estimates | | | | | |
|---|---|---|---|---|---|
| Run Length | Run-In 0 | Run-In 300 | Run-In 660 | Run-In 1380 | Run-In 1500 |
| | | | | ** | |
| 3000 | 1.951 | 2.081 | 2.125 | 2.127 | 2.113 |
| 4000 | 2.023 | 2.124 | 2.159 | 2.169 | 2.164 |
| 5000 | 2.059 | 2.141 | 2.170 | 2.179 | 2.175 |
| 6000 | 2.085 | 2.153 | 2.178 | 2.186 | 2.184 |
| 7000 | 2.109 | 2.168 | 2.189 | 2.197 | 2.196 |
| 8000 | 2.121 | 2.173 | 2.192 | 2.199 | 2.198 |
| 9000 | 2.130 | 2.177 | 2.193 | 2.200 | 2.198 |
| 10000 | 2.137 | 2.178 | 2.193 | 2.199 | 2.198 |
| 11000 | 2.145 | 2.183 | 2.197 | 2.202 | 2.201 |
| 12000 | 2.159 | 2.194 | 2.207 | 2.212 | 2.212 |
| 13000 | 2.174 | 2.207 | 2.219 | 2.225 | 2.225 |
| 14000 | 2.183 | 2.213 | 2.225 | 2.230 | 2.230 |
| 15000 | 2.175 | 2.203 | 2.213 | 2.218 | 2.217 |
| 16000 | 2.176 | 2.203 | 2.212 | 2.216 | 2.216 |
| 17000 | 2.177 | 2.202 | 2.211 | 2.215 | 2.214 |
| 18000 | 2.175 | 2.198 | 2.207 | 2.210 | 2.210 |

**Table 4.2b.** Mean DRYQ queue length estimates as a function of the simulation run length and of the run-in-period.

Other examples showing the performance of our method as compared to Gordon's method as well as to run-in-periods that are longer and shorter than the one estimated here are given in Appendix F.

With the examples discussed in this thesis we show that the run-in-period estimated with the method proposed in this chapter satisfies the measures of performance discussed in Section 4.5.1. In other words the following points are confirmed:

1. The mean estimates obtained with the estimated run-in-period are closer to the real steady state value, $\mu$, than when no run-in-period is used.

2. The steady state is reached for shorter simulation run lengths when the run-in-period estimated with our method is used than if no run-in-period is used.

3. There is usually a smaller difference between the mean estimates and the steady state value, $\mu$, when the run-in-period estimated with the method proposed here is used than when longer or shorter run-in-periods are used.

*A last, but important point to notice is that we have not been looking for an OPTIMAL, but a GOOD approximation for the run-in-period.* Otherwise, it could be argued that as we are comparing the behaviour of the run-in-period estimated with the method proposed in this chapter to only a few other run-in-periods, there may be others with a better behaviour than the one we have estimated. However, we have shown that our estimated run-in-period seems to perform better than longer and shorter run-in-periods (but obviously we are not saying that this is necessarily true for all run-in-periods).

## 4.6. SIMULATION OF SOME SYSTEMS FOR WHICH AN ANALYTICAL ANSWER EXISTS

We have placed special emphasis in the fact that most of the procedures that are proposed by simulation theoreticians lack *generality*. They are usually substantiated by a rigorous mathematical theory, but they are not extensively tested as they tested only with some simple systems with known analytical answer. To overcome this problem we have proposed in this chapter a procedure to deal with the initialisation bias problem and we used it on different simulation models (see example of the LAUNDERETTE in this chapter and additional examples in Appendix F). From the results obtained we can conclude that the run-in-period thus estimated gives good results in the sense that it greatly reduces the simulation run length required for the parameter to reach the steady state and it also gives mean estimates with smaller bias than if other run-in-periods are used. The method proposed here seems to work quite well for complex simulation models where an analytical answer is not available. The next question to answer is if there are limitations to the applicability of this method. To test this point three queuing systems, for which the steady state values can be calculated analytically, were simulated.

These models are:

1. M/M/1 queue.
2. M/M/4 queue.
3. A system of two queues in tandem, which can be analysed using Jackson's theorem.

We give in section 4.6.1. the analytical solution for these models; in section 4.6.2. we make use of the method based on the standard deviation of the sample mean to estimate a run-in-period for the M/M/1 example. Appendix F gives similar results for the other two models.

### 4.6.1. ANALYTICAL SOLUTION FOR THE M/M/s AND FOR QUEUES IN SERIES

The following notation will be used in the discussion of different simple models of queues for which an analytical answer can be obtained:

L = average number of customers present in the system.

$L_q$ = average number of customers waiting in line.

W = average time a customer spends in the system.

$W_q$ = average time a customer spends in line.

$\lambda$ = average number of customers entering the system per unit time.

$\mu$ = average number of customers served per unit time.

$\tau = \lambda/\mu$ = traffic intensity.

For most queuing systems, Little's queuing formula (Little, 1961) can be summarised as follows:

**LITTLE'S THEOREM**

For *any* queuing system in which a steady state distribution exists, the following relations hold:

$L = \lambda W$

$L_q = \lambda W_q$

Making use of this theorem and notation we may explain the following model:

**M/M/1 QUEUE.**

This is a queue where both the arrival and the service processes can be modelled using an exponential distribution. For a steady state solution to exist, the value $\tau = \lambda/\mu < 1$. The value $\lambda$ is known as the **arrival rate** and it has units of arrivals per unit of time. Similarly, the value $\mu$ is called the **service rate** and has units of time per customer. The value $1/\mu$ is called the **mean service time** for a customer.

From queuing theory we obtain the following values for L, W, $L_q$, and $W_q$ for the M/M/1 queue:

$L = \tau/(1 - \tau)$; $\qquad W = L/\lambda = 1/(\mu - \lambda)$

$L_q = \lambda^2/[\mu(\mu - \lambda)]$; $\qquad W_q = L_q/\lambda = \lambda/[\mu(\mu - \lambda)]$;

With these formulas we can calculate the steady state mean queuing time for the customer waiting to be served and the mean queue length in the queue of customers waiting to be served.

**2. M/M/s QUEUE**

This queue is similar to the M/M/1 queue except that the number of servers is s instead of 1. From queuing theory formulas (See Winston, 1987, pp887) we know that the average number of customers waiting in the queue is given by:

$$L_q = P(j \geq s)\tau/(1-\tau) \tag{4.10}$$

where j is the number of customers in the system, and s is the number of servers. $P(j \geq s)$ is given by Eq. 4.11.

$$P(j \geq s) = \frac{(s\tau)^s \, P(j=0)}{(1-\tau)} \tag{4.11}$$

In Eq. 4.11. $P(j = 0)$ is the probability that the system is empty and can be calculated using Eq.4.12.

$$P(j=0) = \frac{1}{\sum_{i=0}^{i=(s-1)} (s\tau)^i/i! + \frac{(s\tau)^s}{s!(1-\tau)}} \tag{4.12}$$

With the value of $L_q$ given by Eq. 4.10., the value of $W_q$ can be calculated using Little's formula.

### 3. QUEUES IN SERIES

In this type of systems the arrival undergoes stage 1 service (after waiting in line if all stage 1 servers are busy on arrival). After completing service in stage 1 the arrival proceeds to stage 2 where if necessary, he waits for service. This process continues until the customer has gone through k stages. This system is called a **k-stage series** and can be analysed using **Jackson's Theorem** (Jackson, 1957).

### JACKSON'S THEOREM

If :

1. Interarrival times for a series queuing system are exponential with rate $\lambda$;
2. service time for each stage *i* server are exponential;
3. each stage has an infinite capacity waiting room,

then interarrival times for arrivals to each stage of the queuing system are exponential with rate $\lambda$.

## 4.6.2. ANALYSIS OF THE RESULTS OBTAINED FOR THE M/M/1 QUEUE

In this section we will apply the method proposed in this chapter to the M/M/1 queue. As the results obtained for the M/M/4 queue and the Jackson system are similar they are given in Appendix F. As discussed in Section 4.4.5., there seem to be some problems in the application of the proposed method to queues with a large value of traffic intensity ($\tau \geq 0.9$). However, this requires further research. With this exception, the procedure here proposed works well not only with complex systems but also with simple systems. Therefore, we feel confident that it will work well for different simulation models.

In the simulation of the M/M/1 queue we use $\lambda = 1/15$, and $\mu = 1/10$. This means that the activity **arrive** follows a negative exponential distribution with mean 15 (i.e., $1/\lambda = 15$). Similarly, the execution time of the activity **service** can be modelled by a negative exponential distribution with mean 10 (i.e., $1/\mu = 10$). Therefore $\tau = (1/15)/(1/10) = 10/15$. Applying the formulas for $L_q$ and $W_q$, we obtain a steady state queue length $L_q = 4/3$ and a steady state waiting time $W_q = 20$.

Figure 4.10. shows the variation of the standard deviation of the mean queuing time estimate for the queue of customers waiting to be served, as a function of the simulation run length. From this graph the estimate of the run-in-period is 300. Using a similar graphical approach for the mean queue length estimates of the same queue, the run-in-period can be taken as 250.

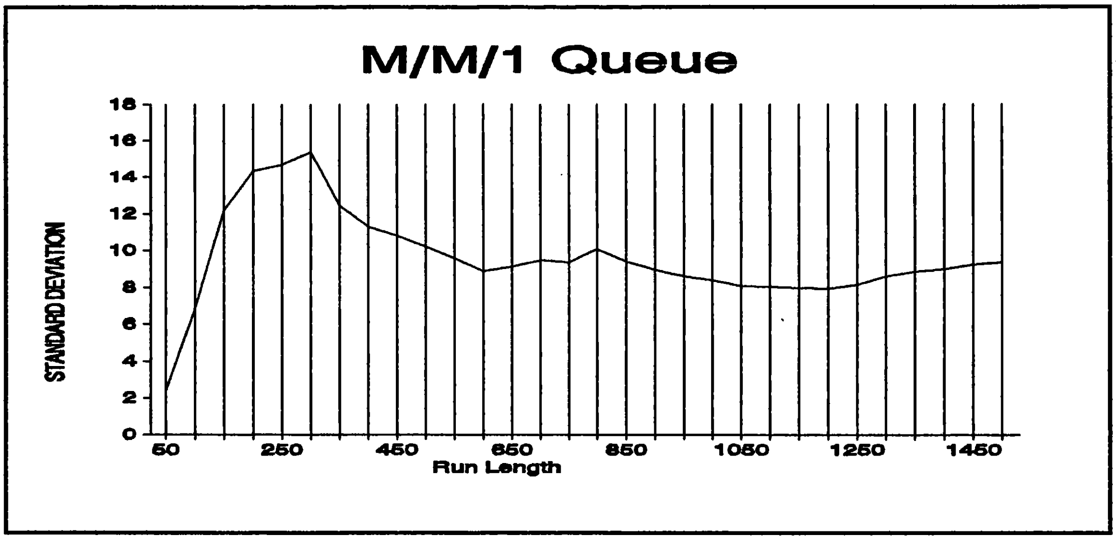



**Figure 4.10.** STANDARD DEVIATION of the mean queuing time estimates for the M/M/1 queue, as a function of the simulation run length.

Tables 4.5a (queuing time) and 4.5b. (queue length) give the mean estimates for the queuing time and the queue length of this queue as a function of the simulation run length and of different run-in-periods. We have underlined the mean estimate for which the parameter can be considered to be in the steady state. i.e., mean estimates with this or longer simulation run lengths all fall within 2.5% of the steady state value.

| M/M/1 Queue - Mean Queuing Time Estimates | | | | | |
|---|---|---|---|---|---|
| Run Length | Run-In 0 | Run-In 100 | Run-In 250 | Run-In 300 | Run-In 450 |
| | | | | ** | |
| 500 | 14.876 | 16.760 | 17.487 | 17.472 | 16.691 |
| 1000 | 17.686 | 18.410 | 18.971 | 19.054 | 19.059 |
| 1500 | 18.335 | 18.855 | 19.251 | 19.301 | 19.343 |
| 2000 | 18.576 | 18.843 | 19.133 | 19.161 | 19.169 |
| 2500 | 18.866 | 19.126 | 19.371 | 19.397 | 19.407 |
| 3000 | 19.047 | 19.260 | 19.473 | <u>19.500</u> | <u>19.509</u> |
| 3500 | 19.085 | 19.293 | 19.477 | 19.515 | 19.509 |
| 4000 | 19.202 | 19.364 | <u>19.526</u> | 19.547 | 19.557 |
| 4500 | 19.350 | 19.443 | 19.588 | 19.607 | 19.618 |
| 5000 | 19.318 | 19.447 | 19.577 | 19.593 | 19.602 |
| 5500 | 19.428 | <u>19.547</u> | 19.667 | 19.682 | 19.692 |
| 6000 | 19.459 | 19.598 | 19.708 | 19.722 | 19.732 |
| 6500 | <u>19.580</u> | 19.715 | 19.819 | 19.833 | 19.845 |

**Table 4.5a.** Mean queuing time estimates for the M/M/1 queue as a function of the simulation run length and for different run-in-periods.

| M/M/1 Queue - Mean Queue Length Estimates | | | | | |
|---|---|---|---|---|---|
| Run Length | Run-In 0 | Run-In 100 | Run-In 250 | Run-In 300 | Run-In 450 |
| | | | ** | | |
| 500 | 1.085 | 1.240 | 1.342 | 1.358 | 1.419 |
| 1000 | 1.249 | 1.313 | 1.362 | 1.368 | 1.378 |
| 1500 | 1.279 | 1.318 | 1.347 | 1.350 | 1.353 |
| 2000 | 1.281 | 1.306 | 1.326 | 1.327 | 1.327 |
| 2500 | 1.292 | 1.313 | 1.330 | 1.331 | 1.331 |
| 3000 | 1.297 | 1.314 | 1.327 | 1.328 | 1.327 |
| 3500 | 1.297 | 1.314 | 1.325 | 1.326 | 1.326 |
| 4000 | 1.301 | 1.314 | 1.324 | 1.325 | 1.324 |
| 4500 | 1.309 | 1.317 | 1.326 | 1.327 | 1.326 |
| 5000 | 1.304 | 1.315 | 1.322 | 1.323 | 1.323 |
| 5500 | 1.310 | 1.320 | 1.327 | 1.328 | 1.328 |
| 6000 | 1.311 | 1.322 | 1.329 | 1.329 | 1.329 |
| 6500 | 1.316 | 1.327 | 1.333 | 1.334 | 1.334 |

**Table 4.5b.** Mean queue length estimates for the M/M/1 queue as a function of the simulation run length and for different run-in-periods.

From the underlined values in Tables 4.5a and 4.5b we conclude that for a confidence of 2.5% the mean queuing time reaches the steady state for a simulation run length of 3000 if the run-in-period is used while it takes up to 6500 units of time when no run-in-period is used.

For the mean queue length the influence of the run-in-period is more noticeable as the parameter is in the steady state for a simulation run length of 500 while when no run-in-period is used it requires a simulation run length of 5500 approximately.As in the case of the LAUNDERETTE, the run-in-periods estimated with the method proposed in this chapter give mean estimates with closer values to the real steady state value and for shorter simulation run lengths.

In Appendix E we approximated a run-in-period of 2500 using Gordon's method. With our method, the mean queuing time requires a simulation run length of 3000 to reach the steady state. The difference between a simulation run length of 2500 (actually somewhat longer as 2500 is the run-in-period and therefore some additional observations have to be recorded) and one of 3000 is not significant and we can consider that our method performs better than Gordon's method as in most of the examples given in this thesis.

## 4.6. CONCLUSIONS

In this chapter a method to deal with the initialisation bias problem is proposed. The method proposed in this thesis for dealing with the initialisation bias problem has several advantages over those already existing: it is simple to use as opposed to the complexity, for example, of Law and Kelton's method; it does not require the setting of parameters that may be model dependent like for example, in Welch's method; there is no need to modify the simulation software as the method is simply based on observations obtained from the simulation when the simulation run length is short; the computer time required for the estimation of the run-in-period is just a small proportion of the total time spent in the simulation project; and just as important, the method has been shown to work well for different simulation models, in the sense that when using it, the steady state is reached for shorter simulation run lengths and the mean estimates are closer to the steady state value than when no run-in-period (or longer or shorter run-in-periods) is used.

The method proposed in this chapter satisfies all those objectives. It is based on the fact that during the transient phase the **standard deviation** of the mean estimates tends to increase rather than decrease when the simulation run length is increased. A good estimate of the run-in-period is given by the simulation run length for which the standard deviation reaches a maximum. Recording data at regular intervals of time, the point for which the standard deviation reaches its maximum value is easily identified with a **line** graph. The method proposed here is much easier to use than those already in existence and, even more important, it is shown that it gives good results for a great variety of simulation models in the sense that the steady state is reached for shorter simulation run lengths than if no observations are deleted. In some cases use of other run-in-periods different to the one estimated with the new method also bring the system to steady state for approximately the same simulation run length. In this case the empirical results show that usually the difference between the mean estimates and the steady state value $\mu$ is smaller for the estimated run-in-period.

It is also important to notice that although the new method is similar to the one proposed by Gordon, in that the deletion of observations is based on an analysis of the variation in the standard deviation of the mean estimate for short simulation run lengths, it does not overestimate the run-in-period as Gordon does, and at the same time the identification of the cut-off point is not easy with Gordon's method while in our method this identification is easy and simple. From a computational point of view the new method is superior to that of Gordon's, because if the slope of the graph is going to be independent of the number of replications, we may require to take a very large number. This value, although the results are not reported in this thesis, is related to the number of replications required to obtain a non-biased estimate. As discussed in Chapter 3, sometimes this number can be as large as 900 or 1000 replications. The only problem, but this also occurs in Gordon's method, is in those cases when, as discussed in Section 4.5.5., due to the large variation of the mean estimate, it may be necessary to increase the number of replications in order to obtain a good approximation of the run-in-period.

Further research is required to extend this method to other parameters estimated from the simulation, and to estimate the run length that is required for the parameter to reach the steady state, as Law and Kelton's method does. Additional research is also required with respect to the problem of the initialisation bias considering alternatives like, for example, starting the simulation in a state more similar to the steady state, or even on using methods for which the standard deviation of the mean estimate is smaller than when the standard method is used. As shown by Grassman, the longer the simulation run length the smaller the standard deviation of the steady state estimate, and as can be expected, the smaller this standard deviation the less the influence of the initial bias (Grassman, 1982).

# CHAPTER 5 : CONFIDENCE INTERVALS IN STEADY STATE SIMULATIONS

## 5.1. INTRODUCTION

Chapter 1 discussed the need for the statistical analysis of simulation output data. However, when using classical statistics techniques to carry out this analysis, the observations obtained from the simulation output should be independent (and identically distributed). To obtain this independence we make use of what is called **Random Numbers**, or in other words, we use **Random Sampling** to simulate a random behaviour (Mihram, 1983).

A problem with this approach is that it gives a low precision of the simulation estimates as the variance of the estimate will sometimes be too large; this creates problems when making inferences about the estimate because most statistical tests performed on this estimate will be "affected" by its large variance and, therefore, it does not give a reliable idea of the real value of the parameter that we are estimating. When a large variance of the estimate makes the simulation of little practical value a technique to reduce this variance is needed. Several variance reduction techniques have been proposed in the literature, like the following:

1. **Antithetic Variables** (Deligonul, (1987),Cheng, (1982); Cheng, (1984) discusses the use of Antithetic Variables in terminating simulations);
2. **Stratified Sampling** (Clark, (1960); Cheng and Davenport, 1988).
3. **Control Variates** (Sharon and Nelson, (1988); Lavenberg and Welch, (1981));
4. **Common Random Numbers** (Heikes et al (1976), Schruben and Margolin, (1978), Wright and Ramsay, (1979));

A good summary of the different techniques for variance reduction can be found in James, (1985), Wilson, (1984) and Law and Kelton (1991); other references are given in the bibliography of this thesis.

The problem with most of these techniques is that "it is not usually clear how they work, and as a consequence, how to use them efficiently." (Saliby, 1990a)

*Descriptive Sampling* (D.S.) is an alternative method to random sampling proposed by Saliby (1990a), and that, as explained in this chapter, may give a better estimate of a parameter in the sense that its variance will be smaller, and therefore any inference on this parameter will be more accurate. The main characteristic of Descriptive Sampling is that it "is based on a deterministic and purposive selection of the input sample values...Contrary to common belief, there is no need for a random selection of values in a Monte Carlo study, or equivalently, there is nothing wrong with a deterministic selection of such values. Once this point is accepted, it becomes evident that a deterministic selection of sample values is the most appropriate approach to be followed in any Monte Carlo application, including Simulation." (Saliby, 1990b).

D.S. has been extensively tested by Saliby using the method of replications, for the estimation of both terminating and steady state estimates. However, it has not been used yet in procedures that estimate the steady state parameters using a single but very long simulation run. As will be seen, due to its characteristics, D.S. seems particularly well suited to be used with the **batch means** method. Conducting research in this field we also identified some important aspects concerning the application of the **batch means** method that are also discussed in this chapter.

### 5.1.1. CHAPTER OBJECTIVES

Throughout the thesis we have pointed out the need for testing procedures not only with simple systems, with known analytical answer, but specially with complex models for which no analytical answer is known. Therefore, one of the objectives of this chapter is to apply the **batch means** method (described in Chapter 1) to some complex simulation models. On doing this we will discuss some of the procedures that have been proposed in the literature to be used with the batch means method.

A second objective of this chapter is to use D.S. with the batch means method. Because of the way it is implemented (Section 5.3.) it will give more accurate estimates than Random Sampling (R.S.).

### 5.1.2. CHAPTER OUTLINE

Section 5.2. gives a theoretical background to Descriptive Sampling. Section 5.3. describes what Descriptive Sampling is, and how to implement it. In Section 5.4 we discuss the Batch Means method and some of the sequential procedures proposed in the literature. In Section 5.5. we discuss empirical results obtained using this method, and we show how some of those proposed in the literature may not work well with complex systems

## 5.2. THEORETICAL BACKGROUND

Systems are often represented as a "black box"; a set of *n* inputs $X_j$ are applied to this box and the "black box" transforms these inputs into a set of *m* outputs $Y_k$. (See Figure 5.1.). This means that the outputs are dependent on the inputs. This is mathematically expressed as :

$$Y_k = F_k(X_1, X_2 ... X_n), \quad k = 1, 2...m. \qquad (5.1)$$

Simulation can also be thought of as a "black box" where the input variables are usually *samples from pre-specified probability distributions* and where the output variables are the results obtained from the simulation. But as the input samples are **randomly** obtained and, even more, the input samples change from replication to replication, we can expect a certain variability in the inputs. This means that the variability in simulation outputs will depend on the input samples variability. Therefore, by studying the variability of the input samples, we get a deeper understanding of the variability that we may expect in the simulation outputs, and hence it may be possible to develop some methods to reduce this variability.

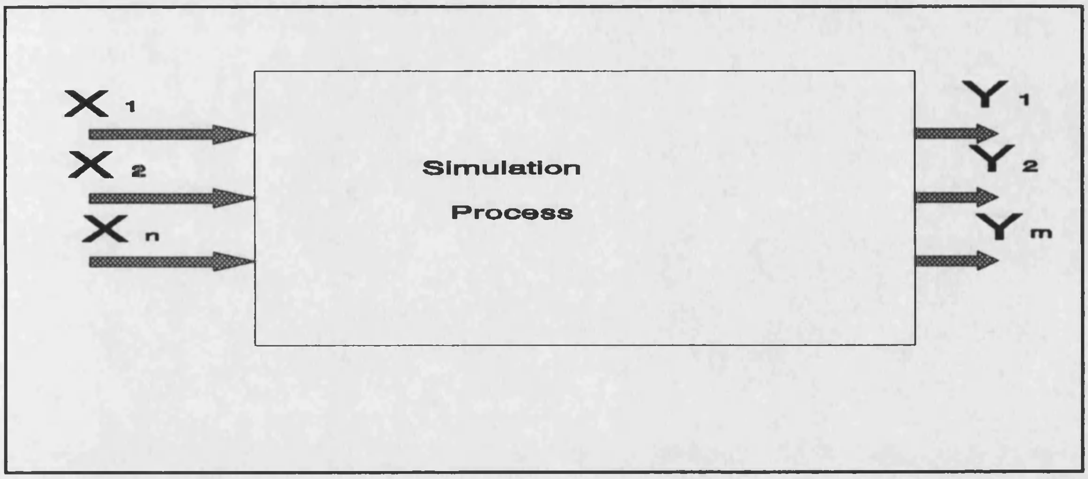



**Figure 5.1.** The simulation process as a "black box".

With this idea in mind Saliby (1990a) found that "in principle, any input sample can be seen as being composed of two global features: the set of input values and their sequence."

We will explain these two concepts with an example. Assume that we use Random Sampling (as usual in simulation practice) to sample from a uniform distribution that takes values between 0 and 1. In a simulation run where 7 of these values are generated, we obtain the following sample:

$U_1$ = [0.51, 0.92, 0.21, 0.34, 0.78, 0.13, 0.98]

For this sample, we define the corresponding SET of values as the sample in ascending order:

$SET_1$ = [0.13, 0.21, 0.34, 0.51, 0.78, 0.92, 0.98]

There are many possible permutations of the values of this set. One possible permutation or SEQUENCE is:

$SEQ_1$ = [4, 6, 2, 3, 5, 1, 7]. What this permutation means is take first the element number 4 of $SET_1$ (0.51); then take the element number 6 of $SET_1$ (0.92), and so on. As may be seen this sequence corresponds to sample $U_1$.

Similarly another permutation or *sequence* of $SET_1$,

$SEQ_2$ = [3, 1, 5, 2, 7, 4, 6]

will correspond to an input sample $U_2$:

$U_2$ = [0.34, 0.13, 0.78, 0.21, 0.98, 0.51, 0.92]

Notice that when using Random Sampling the set varies at random. Notice also that a *sequence* is a *random* permutation of the set values.

But it has been said that the variability of the output depends on the variability of the input samples. By breaking down an input sample into a SET and a SEQUENCE it is possible to study the influence of each one in the variability of the estimate obtained from the simulation output. Continuing this line of thought Saliby (1990a) found that the variability of simulation estimates due to the effect of the set of values can be explained by the deviation of the input sample moments from the corresponding theoretical values. In other words, the input sample is assumed to follow a specific probability distribution. However due to the randomness of the input sample, there is some deviation of the values of this input sample with respect to the theoretical values.

As a conclusion to this study, "the set variability can be considered as a kind of noise which is introduced during the sampling process. Questioning this variability, we derived a new sampling approach in simulation: *Descriptive Sampling*." (Saliby, 1990a.) In summary, in simulation some of the input variables are deterministic, but some others should be sampled from the appropriate probability distribution function. When this sampling is made at random (as it is usually) the input distribution does not closely follow the theoretical distribution it comes from. This will cause an increase in the variance of the estimates obtained from the simulation. This problem is avoided when we use **descriptive sampling.**

## 5.3. DESCRIPTIVE SAMPLING IMPLEMENTATION

As proposed by Saliby (1990a, 1990b), the basic difference between Descriptive Sampling (D.S.) and the most common sampling technique, usually called Random Sampling (R.S.) is that in Random Sampling the input samples are randomly generated while "Descriptive Sampling is based on a deterministic and purposive selection of the sample values - in order to conform as closely as possible to the sampled distribution - and the random permutation of these values. As such, it represents a fundamental conceptual change from Monte Carlo sampling, departing from the "principle" that sample values must be randomly generated in order to describe random behaviour." (Saliby, 1990b).

Therefore, in Descriptive Sampling we use a *deterministic set of values* and we obtain *random permutations* of this set of values every time that a new value is needed. The same *set of values* is used in all the different replications, avoiding in this way the variability that arises in the use of Random Sampling. The only requisite of Descriptive Sampling to obtain a "good" and "close" representation of the theoretical input probability distribution is to know beforehand the input sample size. This means that we have to determine how many times the simulation is going to sample from that particular distribution. For example, the activity **arrive** in the PUB example is sampled from a negative exponential distribution. In order to use Descriptive Sampling, it is necessary to know beforehand how many arrivals to the system will occur for the chosen simulation run length. In some simple simulations this value will be easily determined. But in more complex simulations the determination of this value will not be so straightforward. In this case a few replications are made in order to determine an approximate sample size. The method still works quite well and, from empirical results not given in this thesis, when the simulation run length is long enough, this restriction is not necessary.

The following sub-sections describe the two main steps to take when using Descriptive Sampling.

### 5.3.1. SET VALUES GENERATION.

These values should be generated in advance. They are generated, as in Random Sampling using the general method of *inverse transform*. The only difference is that, because we try to obtain a set of values that is representative of the theoretical probability distribution, these values should "cover" the whole range of the distribution. Therefore if F is the cumulative distribution function and the sample size is **n**, the set values are obtained as

$$x_j = F^{-1}[(i\text{-}0.5)/n], \qquad i=1..n; \qquad (5.2)$$

NOTE: Equation (5.2) follows from the definition of an empirical distribution function if we define the cumulative distribution F(x) as $F(x)=Pr(X \leq x)$, where X is a random variable. "If X has the same distribution as the $X_j$ data, a reasonable approximation to F(x) is thus the proportion of the $X_j$'s that are less than or equal to x. In particular we might define an empirical distribution function $F_n(x) = i/n$, since this is the probability of the $X_j$'s that are less or equal to $X_{(i)}$ (where $X_{(i)}$ is the *i*th. smallest of the $X_j$'s). For purposes of probability plotting, however, it turns out to be somewhat inconvenient to have $F_n(X_{(n)})=1$, that is, to have an empirical distribution function that is equal to 1, for a finite value of *x*. We therefore make a small adjustment and define

$$F_n(X_{(i)}) = \frac{i-0.5}{n} \qquad (5.3)$$

for i=1, 2,...,n." (Law and Kelton, 1991)

**EXAMPLE** The following example shows the generation of the set of values for a uniform distribution taking values between 4 and 10, and a sample size n=50. The formula used to generate uniform distributed variables, by the **Inverse Transform Method**, is the following:

XD[i]=a+((i-0.5)/n)*(b-a)

where a and b are the two end values of the uniform distribution. Using this formula with i=1, 2...50 and a = 4, b = 50 we generate the *set of values* (XD). To obtain independent observations from the simulation output we sample at random from these *set of values.*

### 5.3.2. RANDOM PERMUTATION

Once the set of values has been generated by the procedure described in Section 5.3.1. they "can all be shuffled before carrying out each run, but it is more convenient to shuffle them during the run, drawing a set element whenever is required. In practice, this sequential process is done by sampling the pre-defined set of descriptive values without replacement." (Saliby, 1990b.)

## 5.4. BATCH MEANS METHOD

In sections 5.2. and 5.3. we have discussed a new sampling method called Descriptive Sampling. Its main characteristic is the smaller variance of its estimates as compared to those obtained from random sampling. The rest of the chapter will deal with the **batch means** method, as an alternative method, based on a single long run, for the estimation of steady state parameters. In this section we will describe the method, as well as some of the procedures that have been suggested in the literature to be used when the **batch means** method is employed. This, along with the description of the two previous sections, will give the theoretical background required for the discussion that follows in Section 5.5.

### 5.4.1. USE OF A SINGLE LONG RUN FOR THE ESTIMATION OF STEADY STATE PARAMETERS : GENERAL DISCUSSION.

Several methods have been proposed for the calculation of confidence intervals for steady state estimates. The first of these is the **method of replications** which has already been used and discussed in this thesis; however, the main problem with it is the influence of the initial conditions. If the simulation run length is not long enough to make this influence negligible, or if the deletion of some of the initial observations does not eliminate this influence, the expected value of the mean values $X_i$, obtained from replication i, will not be $\mu$ and therefore the steady state estimate $\bar{X}$ will be biased. Another problem is that when some of the initial observations are deleted the variance of the mean estimate increases and, therefore, the length (width) of the confidence interval (c.i.) will also increase (relative to a fixed simulation run length).

To eliminate the problem due to the influence of the initial conditions some methods based on a *single very long replication* have been developed for the estimation of steady state parameters. Some of these methods are briefly described here:

1. **Batch Means.** A detailed study of this method will be carried out in this section. (See Schriber and Andrews (1979); Law and Kelton (1982a, 1984); Law, (1977, 1983)).

2. **Autoregressive Representation.** This method, developed by Fishman (1973, 1978b) assumes that the process is covariance stationary and can be represented by a *p*th autoregressive method.

3. **Spectrum Analysis.** "This is a complicated technique and is quite expensive due to the large number of covariance estimates which must be computed." (Law, and Kelton, 1984). For further discussion on this method see Fishman (1969, 1973), Heidelberg and Welch (1981a, 1981b), and Duket and Pritsker (1978).

4. **Regenerative Cycles.** This method was developed simultaneously by Crane and Iglehart (1974a, 1974b, 1975a, 1975b) and by Fishman (1973, 1974). The main problem with it is that regenerative points must be identified beforehand, and this may be difficult, and sometimes impossible. (See as well, Cinlar, 1975; Karlin and Taylor, 1975; Iglehart, and Stone, (1983)).

5. **Standardised Time Series.** This method was developed by Schruben (1983) and its principle is to collect a sample of size N which can be considered as a *Time Series* and standardise the *entire time series.* A variant of this method has been recently proposed and, from the results, it seems that the confidence interval estimators obtained (basically *weighted* generalisations of Schruben's standardised time series), perform much better in the small sample size environment. (Goldsman and Schruben, 1990; Glynn and Iglehart, 1990)

When we use a method based on a single long run we are eliminating one problem (influence of the initial conditions) but at the same time we are creating *autocorrelation* of the observations: the observations obtained from the simulation will be correlated and they will in general be non-stationary. If we use classical statistics methods the c.i. thus calculated will have a probability of covering the real value $\mu$ which is much smaller than the theoretical one (1-$\alpha$). This will make the calculation of c.i. for steady state parameters very difficult. (As a matter of fact, although this is a problem which has not been discussed in this thesis, one of the fields of research in the area of simulation is that of obtaining good estimates for the variance of the estimator. For references on this problem, see Chan, and Lewis, (1979), Clark, (1980) and more recently, Glynn and Iglehart,(1988)).

The methods described above use classical statistics techniques to calculate c.i. The main problem with these techniques is that some of their assumptions (i.e., normality) are not met in practice and to overcome this the use of non-parametric statistics has been suggested. Even though this still requires further research, the results obtained so far seem to be quite promising (See Chien, 1988; Kleijnen, 1987)

Some variants of the **batch means** method have been proposed in the literature in order to reduce the autocorrelation of the observations which is the main problem associated with it. Two of these variants are:

1. The **Overlapping Batch Means Method.** (Meketon and Schmeiser, 1984)

2. The **Spaced Batch Means Method.** (Fox et al, 1991)

(See Bibliography for other articles describing these methods).

### 5.4.2. DESCRIPTION OF THE BATCH MEANS FOR STEADY STATE PARAMETERS ESTIMATION.

In the batch means method N observations $X_1, X_2, \ldots ,X_N$ are recorded. These observations are grouped into *b* batches of size N/b and the mean $\bar{X}$ of each one of these batches is calculated by equation 5.4.

$$\bar{X}_i = \sum_{j=1}^{N/b} \frac{X_{j+(i-1)N/b}}{N/b} \qquad (5.4)$$

The mean of the b batches will give the estimate $\bar{Y}$ (grand batch mean) for the parameter(s) of interest, as shown in equation 5.5.

$$\bar{Y} = \sum_{i=1}^{b} \frac{\bar{X}_i}{b} \tag{5.5}$$

The main problem with this method is to choose a batch size sufficiently large such that successive batches are independent. If they are correlated the variance estimator will be biased (either positively or negatively) and therefore the c.i. thus calculated will be either too short or too large.

### 5.4.3. SEQUENTIAL PROCEDURES

When we want to obtain a c.i. for a steady state parameter we may fix beforehand the simulation run length and, when using the batch method, obtain a number b of independent batches. An important step in this case is to carry out a test for autocorrelation of the batch means. Several methods have been proposed in the literature. Appendix H describes a test based on the Von Neumann statistic and which was used to test the batches for the presence of autocorrelation in the experiments reported in this thesis. (See also Fishmann, 1978a). From the mean of these b different batches we calculate a c.i. for the parameter of interest. However, if the simulation run length is too short some problems have been identified in the literature:

1. The estimate obtained may still be part of the transient response. In other words, if $\bar{X}_i$ is the mean value of batch i, it is possible that for a short simulation run length $E(\bar{X}_i) \neq \mu$ and therefore the point estimate will be biased and the c.i. will be calculated around the wrong value.

2. The variance and therefore, the standard deviation of the estimate may be too large. This means that the c.i. half width length, and the c.i. relative half-width length, will be too large to be of practical use.

3. It may not be possible to find a batch sample size such that the different batches are independent.

**NOTE.** The c.i. relative half width, also called, **c.i. relative precision,** is the c.i. half width divided by the mean estimate. The **c.i. absolute width** is simply the c.i. half width.

In order to deal with these problems the use of a sequential approach is recommended. In this approach we start with an initial batch size, test the batch means for correlation and if they are uncorrelated then calculate a c.i. If the c.i. relative half width (in some cases the absolute half width) is too large or if there is some autocorrelation, then the batch size is increased.

To illustrate the sequential approach we describe one of the methods proposed in the literature by Law and Carson.

## LAW AND CARSON'S METHOD

"The goal of the Law and Carson sequential procedure which is based on the method of batch means, is to construct a $100(1-\alpha)\%$ c.i. for $\mu$ with a relative precision of $\gamma$." ($\gamma$ is the c.i. half width divided by the mean estimate; it is also called the c.i. *relative precision).* "Suppose that $m$ observations from a single simulation run are available. The procedure divides these m observations into 400 batches of size $k$ ($m$ is assumed to be divisible by 400). If the estimated lag 1 correlation between the resulting 400 batch means is less than a threshold value c=0.4, then the same $m$ observations are divided into 40 batches of size $10k$ and the corresponding 40 batch means are considered to be uncorrelated. This indirect approach is necessary to obtain a precise correlation estimate. The 40 batch means are used to obtain a c.i. for $\mu$ using the usual batch means approach, and if the relative precision is not less than $\gamma$, then this c.i. is accepted. If the estimate lag 1 correlation is less than 0.4 or if the actual relative precision is not less than $\gamma$, then $m$ is increased and the above steps are repeated. Law and Carson (1979) recommend that the procedure be applied with the value of $\gamma$ chosen to be less or equal to 0.075." (Law, 1983)

Other methods that have been proposed for the estimation of confidence intervals are the Mechanic and McKay procedure (Mechanic and McKay, 1966) and a procedure proposed by Adams (1983). The objective in Mechanic and McKay's procedure is the calculation of a valid c.i., while in

Adams' procedure the objective is the calculation of a preassigned confidence interval, but to this respect, "further work needs to be done to investigate the performance of this method when applied to other types of stochastic systems, e.g., inventory systems, and job-shop type of systems". (Adam, 1983)

A good survey of both fixed-sample size and sequential procedures for the calculation of steady state confidence intervals is given in Law and Kelton, (1984, 1982a).

### COMMON FEATURES OF THE SEQUENTIAL METHODS PROPOSED IN THE LITERATURE

All the methods named above, as well as some others not discussed in this thesis, including procedures proposed to be used with the regenerative method (See Chapter 1), use as a stopping criterion the c.i. absolute or relative precision. A second feature common to these methods is that they are tested only for simple models with little interaction among their entities.

We will show how in practical simulations use of a stopping criterion related to the c.i. half-width works well, but we will also show how some other conclusions that have been obtained using simple models do not equally work well when applied to more typical simulation models.

## 5.5. DISCUSSION OF THE BATCH MEANS METHOD.

In section 5.5.2. we discuss some points concerning the use of the batch means method when it is applied to some complex simulation models that do not have an analytical solution. In section 5.5.3. we describe the different experiments that were performed to illustrate the points of section 5.5.2. The empirical results obtained from these experiments are given and analysed in section 5.5.4.

### 5.5.1. MODIFICATION OF THE BATCH MEANS METHOD

Due to the type of simulation software used in the experiments reported in this chapter, the batch means method has been slightly modified. Instead of recording N observations and grouping them into b batches, a simulation run length $T_0$ is chosen. This simulation run length should be sufficiently long for the parameter to reach the steady state. Then, this run length is divided into b equal intervals with a sub-run length of $T_1 = T_0/b$. The batch means $\bar{X}j$ (j=1, 2,..b) are obtained as the average of the observations recorded during the interval of time $[(j-1)*T_1, j*T_1]$. We need to test these batch means for autocorrelation, and if necessary, we will have to increase the period of time $T_1$. The estimate $\bar{Y}$ is given by the average of these batch means $\bar{X}_j$, as shown in Equation 5.6.

$$\bar{Y} = \sum_{j=1}^{b} \frac{\bar{X}_j}{b} \tag{5.6}$$

Therefore, with this modification, the batch size defined before as the number of observations in each batch becomes a random number but the method still works well. We employ the same terminology as with the batch means method and the term "batch size" refers to the sub-run length $T_1$.

### 5.5.2. POINTS RELATED TO THE APPLICATION OF THE BATCH MEANS METHOD TO COMPLEX SIMULATION MODELS

As pointed out in Chapter 2, the use of empirical research helps to highlight aspects that are common to different simulation models. We have also emphasised the importance of applying any proposed procedure to systems that show more interaction amongst their entities than simple systems like the M/M/1 queue. By doing this we have been able to identify the following problems and interesting aspects concerning the use of the batch means method:

1. From the empirical results we will show how there exists a minimum total simulation run length, independent of the batch size, that is required for a given parameter to reach the steady state.

2. D.S. **batch mean** estimates will have a smaller variance and for this reason they will require shorter simulation run lengths for the c.i. relative precision to be smaller than a pre-assigned value c′.

3. Concerning the number of batches, Schmeiser (1982), and Law and Carson (1979) suggest that no more than 30 or 40 batches should be used. For example, to this respect Schmeiser in his study of the effects of the number of batches in the analysis of the simulation output says that "our most fundamental conclusion is that $10 \leq k \leq 30$ (where *k* is the number of batches) is reasonable for most simulations." And, as described in Section 5.4.3., in Law and Carson's method, a number of 400 batches are obtained and tested for autocorrelation, and then the observations are regrouped again to form only 40 batches in order to obtain the mean estimate and the c.i. Similarly, the method described in Appendix H, based on the Von Neumann statistic to test for autocorrelation requires at least 100 batches. However, as we will show, the number of batches has very little influence with respect to the accuracy of the batch mean estimate and the c.i. relative precision, and therefore, there is no need of regrouping the observations to form a smaller number of batches after the test for autocorrelation has been carried out.

### 5.5.3. DESCRIPTION OF THE EXPERIMENTS

In order to illustrate the points discussed in section 5.5.2. we carried out some experiments that are described in this section. We used different batch sizes, different number of batches and different sampling methods, as is explained below.

A long simulation run length was chosen (For example 15,000,000 for the LAUNDERETTE model studied in this chapter). In each case different subrun lengths were used; the value of these subrun lengths is such that 10, 20, 30, 50 and 100 batch means are obtained for each one of the parameters of interest (queuing time or queue length of some of the queues of the model).

The estimate $\bar{Y}$ is calculated from Equation 5.6., where $\bar{X}i$ is the value of the mean in batch i and b is the number of batches (10, 20, 30 etc.). For each estimate the standard deviation, and the confidence interval relative precision (at a 95% confidence level) were also calculated.

One of the important points we show using empirical results is that there is a simulation run length R* such that for simulation run lengths shorter than this critical value the system does not reach a steady state. To check this point in an empirical way we determined the batch mean corresponding to b batches (b=10, 20 etc.) and also the batch mean corresponding to b1 batches, with b1 taking values 2, 3...b. The batch mean in this case is easily calculated from Equation 5.7.

$$\bar{Y}(b1,b) = \frac{\sum_{i=1}^{b1} \bar{X}_i}{b1} \tag{5.7}$$

Calling $T_0$ the total simulation run length (15,000,000 for the model studied in this chapter), $\bar{Y}$(b1,b) may be interpreted as the batch mean estimate if b1 batches are recorded, and where each batch has size $T_0$/b.

### 5.5.4. ANALYSIS OF THE RESULTS OBTAINED FOR THE LAUNDERETTE MODEL.

We obtained the mean queuing time and the mean queue length estimates for the following queues of this model:

1. WASHQ;
2. BIDLE;
3. DRYQ;
4. WMIDLE.

Using the results obtained for these queues we will discuss in detail each one of the points of section 5.5.2.

**1. Minimum simulation run length required for the parameter that is being estimated to reach the steady state.**

In Chapter 3 we showed how by increasing the number of replications it is possible to obtain a good approximation to the real, but unknown, curve of the mean estimates as a function of the simulation run length. In this way it is possible to estimate the simulated time for which the curve becomes horizontal. A similar graphical approach can be used with the batch means method to determine the simulation run length for which the curve becomes horizontal. However, in this case this depends only on the total simulation run length. To show this, the LAUNDERETTE model was simulated using different random number seeds. Figure 5.2. shows the batch mean estimates calculated using Equation 5.7. and corresponding to the mean queuing time parameter of the WASHQ.

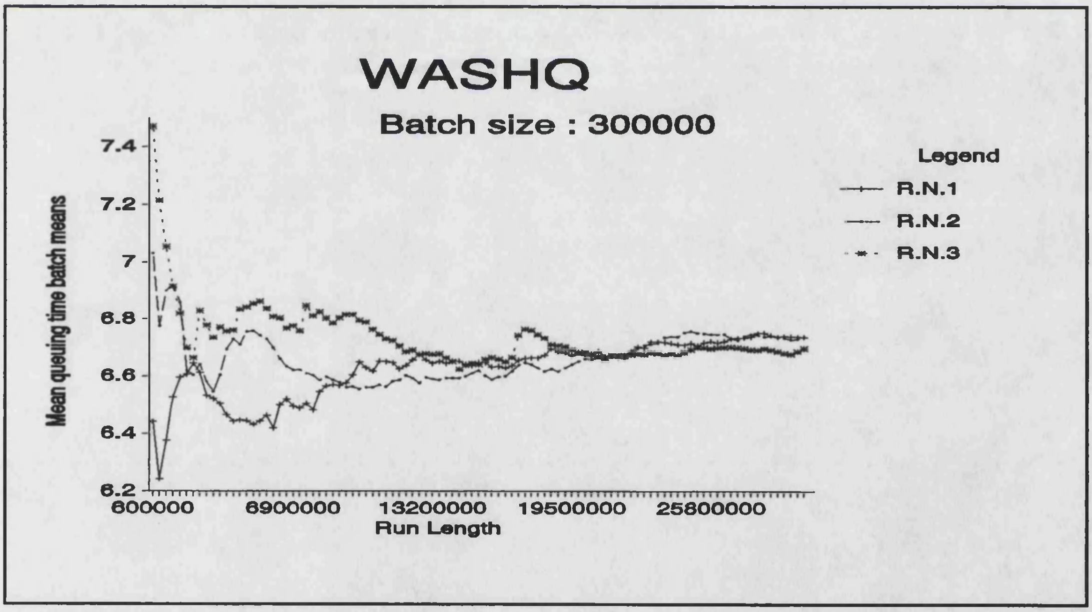



**Figure 5.2.** WASQ mean queuing time batch estimates obtained using different random number seeds.

From this graph we note that they converge to a value of 6.67 and that to obtain this convergence a minimum simulation run length of 13,200,000 units of time is required. In practice it is not necessary to obtain an estimate so close to the real steady state value, and a certain tolerance is allowed. If this tolerance is expressed as a percentage of $\mu$, the real but unknown steady state value, we can use the c.i. relative precision as a stopping criterion. Based on this stopping criterion we describe in detail the algorithm that should be followed in order to estimate steady state parameters in Section 5.6. This algorithm can be summarised as follows: once a batch size for which the batch means are uncorrelated has been identified, we obtain one batch at a time, update the c.i. relative precision, I, and compare it to the c.i. relative precision $c'$, chosen a priori. If $I \leq c'$ then we stop; otherwise an additional batch mean is obtained. This procedure is continued until $I \leq c'$.

Figure 5.3. illustrates this procedure and how the use of the c.i. relative precision as a stopping criterion gives accurate mean estimates. This figure shows the batch mean estimates as a function of the simulation run length and, using a second Y-axis, we also give the corresponding c.i. relative precision. Also shown in this graph are the limits of the c.i. corresponding to simulation run lengths of 5,100,000, 7,500,000, 9,900,000, 12,300,000, and 14,700,000. We note how in all the cases the c.i. "covers" the real value. this value, which has been approximated in an empirical way in Appendix C, is 6.675.

**2. D.S. BATCH MEAN estimates will require shorter simulation run lengths for the c.i. relative precision to be smaller than a pre-assigned value $c'$.**

Using different batch sizes we obtained the batch mean estimates as a function of the number of batches (as described in Section 5.5.3.), for both D.S. and R.S. The information is all similar, and we have summarised it in a graphical way by showing the queuing time **batch mean** estimates for the DRYQ as a function of the batch size and of the simulation run length (which is related to the number of batches given a fixed batch size).

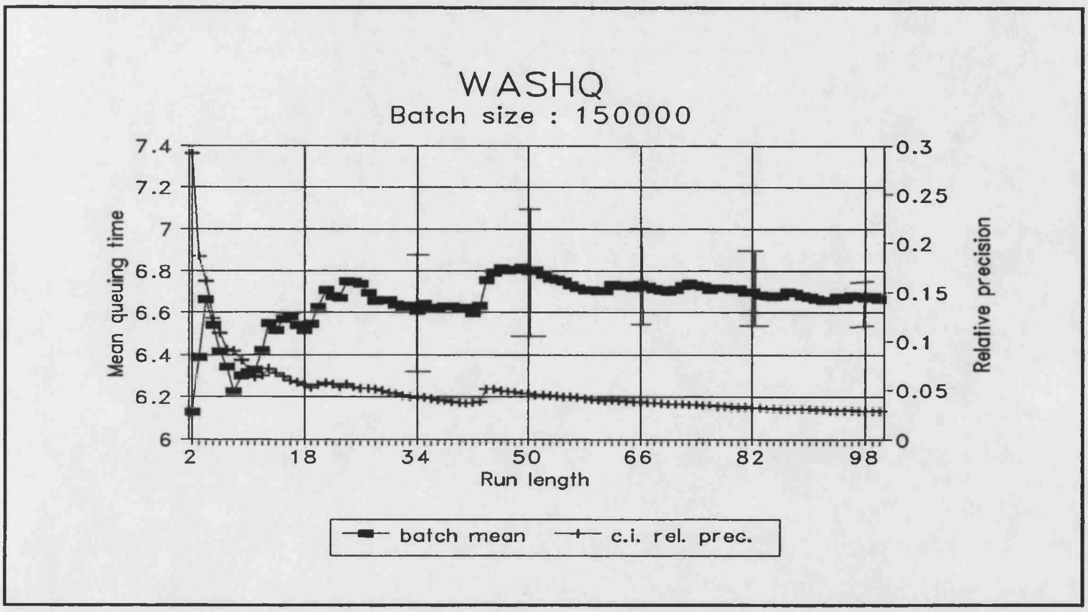



**Figure 5.3.** WASHQ batch queuing time mean estimates, and corresponding c.i. relative precision. Limits for the c.i. obtained for different simulation run lengths are also shown.

Figures 5.4. (batch size : 1,500,000), 5.5. (batch size : 500000) and 5.6. (batch size : 150000), show these mean estimates. From these figures we notice that independent of the batch size, the D.S. batch mean estimates converge to the steady state for shorter simulation run lengths (i.e., there is a smaller change in the value of the mean estimates as the simulation run length increases as compared to the change in the R.S. mean estimates). This implies that it is possible to use shorter run lengths and obtain mean estimates as accurate and close to the real steady state value, $\mu$, if D.S. is used than if R.S. is used.

In order to give an idea of the variation in the value of the standard deviation when the two sampling methods are used, Figure 5.7. shows this variation as a function of the simulation run length for both D.S. and R.S. batch mean estimates, and for a batch size of 1,500,000.

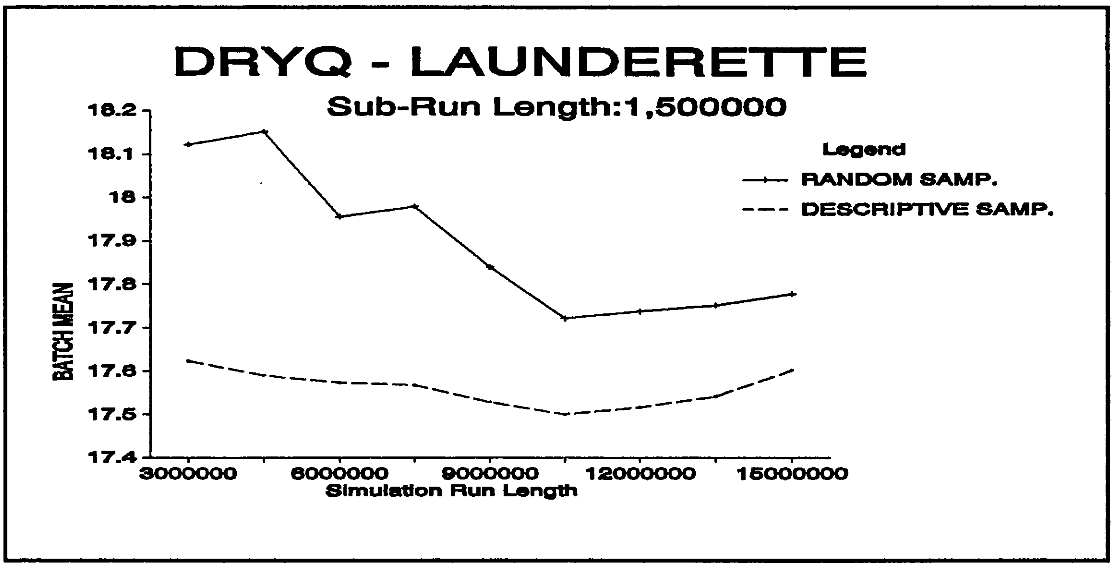



Figure 5.4. Variation in the R.S. and the D.S. batch mean estimates as a function of the simulation run length. The total simulation run length has been divided in 10 batches.

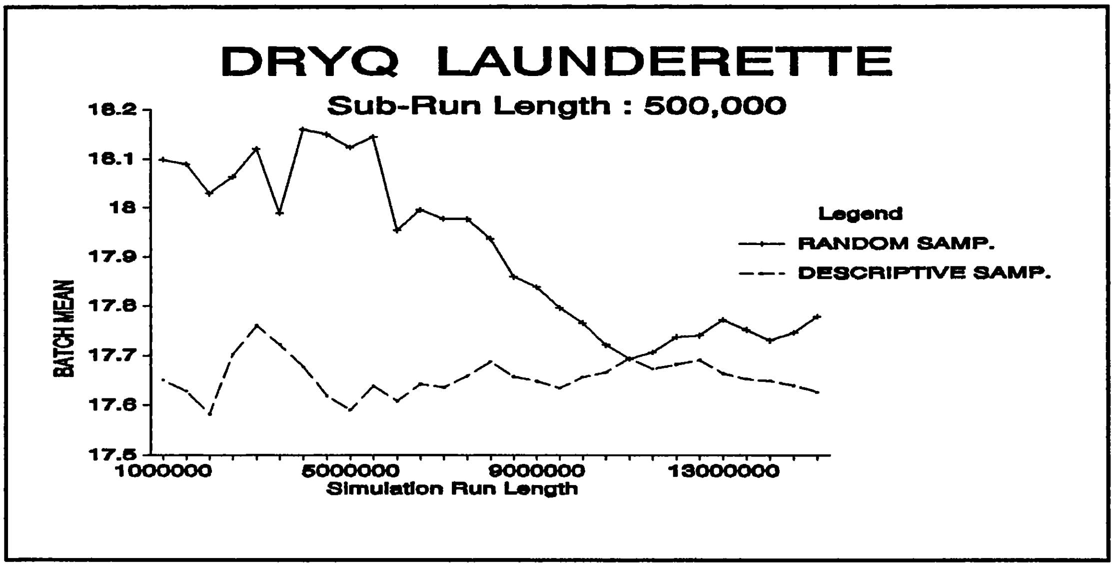



Figure 5.5. Variation in the R.S. and the D.S. batch mean estimates as a function of the simulation run length, when the total run length has been divided in 30 batches.

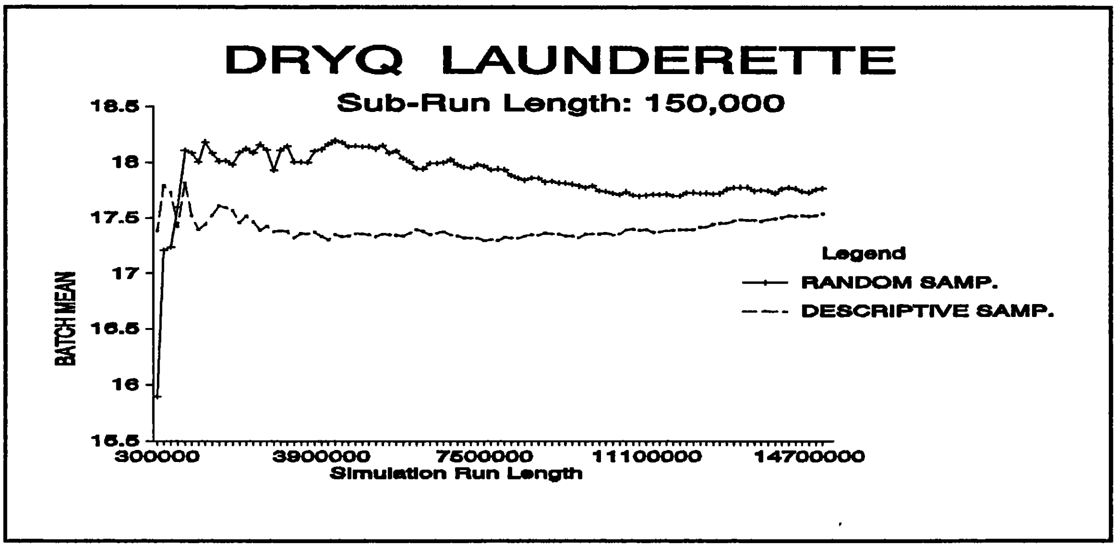



Figure 5.6. Variation in the R.S. and the D.S. batch mean estimates as a function of the simulation run length, when the run length has been divided in 100 batches.

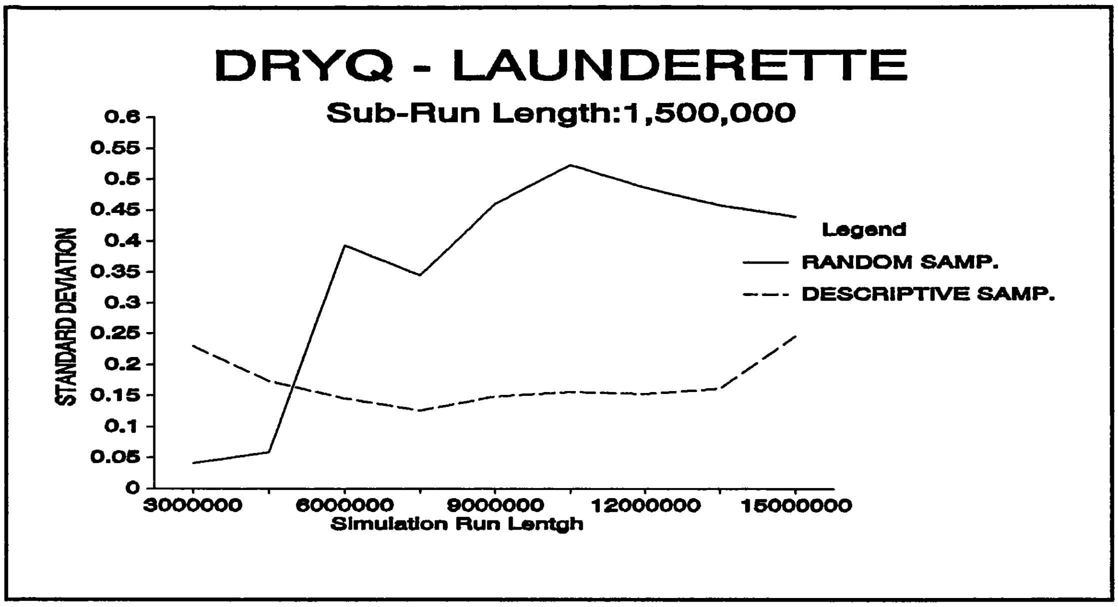



Figure 5.7. STANDARD DEVIATION corresponding to the batch mean estimates of Figure 5.4.

As we would expect, the standard deviation for the D.S is smaller and therefore the c.i. half width is shorter; as the point estimate, when the parameter has reached the steady state, is approximately the same, the D.S. c.i. relative precision will be smaller.

Therefore, the simulation run length required for the c.i. relative precision to be less than or equal to c' is shorter than the corresponding simulation run length when R.S. is used.

### 3. Influence of the number of batches.

In table 5.1 we summarise some of the points that have been discussed with respect to the use of the **batch means** method, and we show that the batch mean estimate obtained for a given total simulation run length, as well as the c.i. relative precision, are independent of the number of batches.

In this table we give the batch mean estimates for a total simulation run length of 15,000,000, for different batch sizes and for both R.S. and D.S. We also include in this table the values of the 95% c.i. relative precision for each one of the different batch sizes. In order to give an idea of how the **batch means** method compares to the **replications** method we also include the results obtained for the queues of this model, when 300 replications are used and the total simulation run length is 120000.

From the table we may conclude (and this conclusion is true in all the other models studied in this chapter) that for a total simulation run length of 15,000,000 it is valid that:

1. The point estimate (mean queuing time or mean queue length) is approximately the same, independent of the number of batches.

2. The relative precision of the confidence interval is approximately the same for a given parameter, independent of the number of batches.

3. When the simulation run length is very long, and it needs to be very long for the point estimate not to be biased, the c.i. half-width and the c.i. relative precision are smaller than when the replications method is used.

4. Comparing the **batch mean** results with the steady state values ($\mu$)

obtained from the results of Appendix C and reproduced in point 1 of this sub-section, the Descriptive Sampling batch means estimates are closer to the real steady state value than the Random Sampling estimates. In most cases the percentage error of the Descriptive Sampling batch mean estimates is almost zero.

### 5.5.5. SUMMARY OF OUR RESULTS CONCERNING THE USE OF THE BATCH MEANS METHOD

*Based on empirical results (Section 5.5.4. and Appendix G), obtained for some typical simulation models, the research reported in this thesis shows some new important aspects concerning the use of the* ***batch means*** *method (Section 5.5.2). At the same time, the use of complex models shows why some techniques widely used in the* ***batch means*** *method do not always work well in practice, in the sense that they are not general enough to be used for different types of problems, and therefore cannot be applied to simulation models with different characteristics.*

## 5.6. ALGORITHM

In Section 5.5. we confirmed, based on empirical results obtained for more typical simulation models, that the c.i. relative precision can be used as a stopping criterion in a batch means sequential method. It was also shown that the number of batches is not critical as long as the batches are uncorrelated and that the batch means follow approximately a normal distribution.

In this section we will summarise the procedure to be followed for the estimation of steady state parameters using the batch means method.

Based on these facts we propose the following algorithm:

1. Select a large simulation run length, $T_0$. Divide this run length into sub-runs lengths of $T_0/100$. (The value of 100 batches as minimum is suggested by Kleijnen).

| RANDOM SAMPLING | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Queue | Parameter | 10 Batches | | | 30 Batches | | | 100 Batches | | |
| | | Batch | Std. Dev | C.I. Half | Batch | Std. Dev | C.I. Half | Batch | Std. Dev | C.I. Half |
| WASH | Queue. | 6.704 | 0.256 | 0.027 | 6.703 | 0.474 | 0.026 | 6.700 | 0.856 | 0.025 |
| WASH | Queue | 0.839 | 0.033 | 0.028 | 0.839 | 0.061 | 0.027 | 0.839 | 0.112 | 0.026 |
| BIDLE | Queue. | 67.141 | 0.657 | 0.007 | 67.144 | 1.094 | 0.006 | 67.156 | 2.156 | 0.006 |
| BIDLE | Queue | 8.399 | 0.063 | 0.005 | 8.399 | 0.108 | 0.005 | 8.399 | 0.209 | 0.005 |
| DRYQ | Queue. | 17.779 | 0.439 | 0.018 | 17.778 | 0.774 | 0.016 | 17.771 | 1.477 | 0.017 |
| DRYQ | Queue | 2.224 | 0.060 | 0.019 | 2.224 | 0.104 | 0.017 | 2.224 | 0.200 | 0.018 |
| WMIDLE | Queue. | 12.771 | 0.150 | 0.008 | 12.772 | 0.247 | 0.007 | 12.775 | 0.508 | 0.008 |
| WMIDLE | Queue | 1.598 | 0.015 | 0.007 | 1.598 | 0.025 | 0.006 | 1.598 | 0.051 | 0.006 |

| DESCRIPTIVE SAMPLING | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Queue | Parameter | 10 Batches | | | 30 Batches | | | 100 Batches | | |
| | | Batch Mean | Std. Dev | C.I. Half W. | Batch Mean | Std. Dev | C.I. Half W. | Batch Mean | Std. Dev | C.I. Half W. |
| WASH | Queue. Time | 6.586 | 0.196 | 0.021 | 6.631 | 0.417 | 0.024 | 6.528 | 0.714 | 0.022 |
| WASH | Queue Length | 0.824 | 0.025 | 0.021 | 0.830 | 0.052 | 0.024 | 0.817 | 0.089 | 0.022 |
| BIDLE | Queue. Time | 67.323 | 0.247 | 0.003 | 67.284 | 0.392 | 0.002 | 67.395 | 1.002 | 0.003 |
| BIDLE | Queue Length | 8.422 | 0.031 | 0.003 | 8.418 | 0.047 | 0.002 | 8.430 | 0.119 | 0.003 |
| DRYQ | Queue. Time | 17.603 | 0.246 | 0.010 | 17.626 | 0.369 | 0.008 | 17.541 | 0.938 | 0.011 |
| DRYQ | Queue Length | 2.202 | 0.031 | 0.010 | 2.205 | 0.047 | 0.008 | 2.194 | 0.118 | 0.011 |
| WMIDLE | Queue. Time | 12.783 | 0.023 | 0.001 | 12.768 | 0.061 | 0.002 | 12.798 | 0.137 | 0.002 |
| WMIDLE | Queue Length | 1.599 | 0.003 | 0.001 | 1.598 | 0.007 | 0.002 | 1.601 | 0.014 | 0.002 |

| REPLICATIONS METHOD (RANDOM SAMPLING) | | | |
|---|---|---|---|
| Queue | Parameter | Sample Mean | Std. Dev |
| WASH | Queue Time | 6.653 | 1.018 |
| WASH | Queue Length | 0.832 | 0.131 |
| BIDLE | Queue. Time | 67.349 | 2.278 |
| BIDLE | Queue Length | 8.417 | 0.233 |
| DRYQ | Queue. Time | 17.671 | 1.694 |
| DRYQ | Queue Length | 2.209 | 0.225 |

**Table 5.1.** Mean queuing time and mean queue length estimates using the batch means (R.S. and D.S.) and the replications method; the total simulation run length for the batch mean estimates is 15,000,000 and it is divided into 10, 30 and 100 batches; the total simulation run length for the method of replications is 120000.

2. Obtain 100 batches, each one with batch run length of $T_0/100$.

3. Test for autocorrelation of the batch means using Von Neumann's statistic (See Appendix H).

4. If the batch means are uncorrelated, test for normality of the means (See NOTE "a" below). However, it seems (Law, (1980)) that non-normality of the observations does not have a large effect on the estimation of a c.i. (This point requires further investigation, because as shown here some of the results proposed in the literature do not work well when used with complex systems).

5. If the batch means are correlated or if they are highly non-normal, increase the simulation run length to say , $2T_0$; repeat steps 3 and 4.

6. Once a simulation run length for which the batch means are uncorrelated has been estimated, calculate the grand mean $\bar{Y}$ using Eq. 5.6, and the c.i. relative precision.

7. If the relative precision, I is less than or equal to a pre-specified value $c'$ then stop the simulation and perform an statistical analysis of the simulation output. Otherwise, obtain an additional batch mean, update the value of I and compare this value to $c'$. This will be repeated until $I \leq c'$.

**NOTES**

a. Several procedures have been proposed to test a data set for normality. Among these tests we have Shapiro and Wilk's W statistic (1965), and its extension proposed by Royston (1981); Stephens (1974) provides a good summary and comparison among different goodness of fit tests that can be used to test data for other distributions besides the normal distribution. Liliefors (1967) discusses the Kolmogorov-Smirnov statistic and generalises it to make it useful when the mean of the distribution is unknown.

b. Some experiments to deal with the problem of autocorrelation were performed using a similar method to the **spaced batch means method**. The results of these experiments are not reported here but in principle, if $\bar{X}_1$, $\bar{X}_2 ... \bar{X}_m$ are the batch means, we can obtain a mean estimate by considering:

$$\bar{Y} = \sum_{i=1}^{m/2} \frac{\bar{X}_{2*i-1}}{m/2} \tag{5.8}$$

This means that we will have "gaps" in the batch means used to obtain the steady state estimate. Although further research is required it seems to perform quite well, as the batch size could be smaller and still give independent observations.

c. Usually the queues that are more directly influenced by a negative exponential input distribution will take longer to reach the steady state, and the approach to it may be extremely slow (see for example the mean values recorded for the WAIT queue in the PUB model in Appendix G).

d. From the empirical results given in this chapter (and in Appendix G) D.S. should be preferably used when the batch means method is used for the estimation of steady state parameters as it gives estimates closer to the steady state; at the same time, the simulation run length required to obtain a c.i. relative precision smaller than a pre-assigned value c' is shorter if D.S. is used than if R.S. is used.

e. It was shown how the variation in the D.S. estimates is much smaller than that of the R.S. estimates; this is confirmed with the results of Appendix G. Therefore, another important advantage of D.S. over R.S. is that the time required to collect the data from the simulation can be considerably smaller.

## 5.7. CONCLUSIONS

The main topic of this chapter has been the batch means method. On applying it to some more typical simulation models than those usually used by the simulation theoreticians, we have identified some problems and important aspects concerning its application.

D.S. performs particularly well when used with the **batch means** method, and shorter simulation run lengths are required to get an accurate estimate of the parameter of interest. This is due to the way D.S. is implemented. As explained in section 5.2.3. it generates a set of deterministic values, and uses them throughout the simulation. In order to generate this set of values the sample size should be known in advance. However, we found in this research that when the sample size is large enough (we used a maximum of 7500 for the sample size) accurate estimates are obtained even if in theory we should use a larger sample size. At the same time, the batch means method obtains estimates for a given batch size. Therefore, we found that we can generate a set of values using a maximum sample size and sample them at random for each different batch. Use of a maximum sample size, that may be considerably smaller than what would be required, is possible due to the way D.S. has been programmed in practice. Sampling is done without replacement until all the set of value has been used. If additional sampling is required then the same set of values is used again. For this reason, we said in Section 5.1. that D.S. seemed particularly well suited to be used in the **batch means** method.

While in the previous chapters we have obtained steady state estimates based on the method of replications, in this chapter this estimation has been done using a single, but very long simulation run length. Which method to use is more a decision of the practitioner. While the method of replications has the problem of the initialisation bias, the batch means method presents the problem of the autocorrelation of the observations. In either method the simulation practitioner needs to select the simulation run length. When the method of replications is used we showed how it is possible to estimate the number of replications such that the curve of the variation of the mean estimates as a function of the simulation run length is a close approximation to the real one. This will permit the identification of the approximate simulation run length for which the curve becomes horizontal. But, there is an additional problem which the practitioner has to deal with and that is the problem of the initialisation bias. Therefore, in most simulations, after the number of replications that should be used has been estimated, the practitioner needs to identify the number of initial observations to be deleted (Chapter 4) or similarly, use the weighted averages method which is described in Chapter 6. It seems then that the method of replications requires two previous steps before the simulation for the estimation of steady state parameters is run: estimation of the number of replications and estimation of the number of replications to be deleted. When the batch means method is used, we showed in this chapter how the simulation should be continued until the c.i. relative precision is less than or equal to a pre-specified value c'. In this sense the batch means method is simpler to use than the replications method. From the examples showed in this chapter (and in Appendix G) it is clear that D.S. should be used in preference to R.S. as the variation in the batch means as the simulation run length increases is considerably smaller. In both cases, replications and batch means method, it is important to carry out an statistical analysis of the simulation output data as discussed in Chapter 1. A simple, but good way of doing this is by obtaining the c.i. width, and the c.i. relative precision of the mean estimate.

A point that requires more research is that of the simulation run length that is required in order to obtain an "accurate" estimator: when using the method of replications, if the expected value of the observations $X_i$ is different from the real value of the parameter we are estimating, $\mu$, then the c.i. is calculated around a value different to the steady state value. When using the batch means method the main problem is that of avoiding correlation among the batch means. Some procedures have been suggested in the literature concerning the optimal simulation run length or equivalently, the number of observations $X_i$ required to obtain an accurate estimate. Information about some of these procedures can be found in Heidelberg and Welch (1983), Adlakha and Fishman (1982), Fishman (1971), and Robinson, (1976). However, the proposed procedures have all been tested using simple systems, for which an analytical answer exists and as shown in this chapter in practice the performance of such procedures may not be as good as expected. For example, based on experiments performed with the M/M/1 queue and alike, it has been suggested that the number of batches should be less than 30. But, as shown in the examples of this chapter, the mean estimate and the c.i. relative precision depend on the total simulation run length and not on the number of batches.

The methods studied here deal with the calculation of confidence intervals for means. However, simulation should not be limited only to the use of sample means. As a matter of fact, the literature shows that some research has been done on the calculation of confidence intervals for percentiles and proportions. It seems from the results obtained in these studies that, as long as the sample size is large enough so that asymptotic normal theory can be used, a proportion statistic is widely applicable and easy to use (See Mamrak and Amer, (1980)). A topic of further research is to extend our results (and this is valid not only for this chapter but also for the previous chapters) to the use of proportions; considering that we have shown for example the advantages of the use of D.S. over R.S. for the estimation of mean parameters, its use should also give good results when applied to more general simulations.

Another topic of further research concerning the area of confidence intervals is that of the calculation of confidence intervals for the variance parameter. New methods are proposed in the literature, where "the new point and interval estimators for the variance parameter are compared to the classical batch estimator. The results show that the new estimators have asymptotic properties, that clearly dominate the classical estimator." (Chen, 1990).

# CHAPTER 6 : ALTERNATIVE METHODS OF EVALUATING STEADY STATE PARAMETERS

## 6.1. INTRODUCTION

One of the problems, identified in Chapter 4, with the existing methods for dealing with the initialisation bias problem is that they require the setting of parameters that may be model and parameter dependent. We confirmed this point in Chapter 5 by applying the batch means method to complex simulation models. Even in the procedure proposed in Chapter 4 for the estimation of a run-in-period there may be some problems when the parameters for which a run-in-period is to be estimated have a large variance. To avoid the problem of setting or even of estimating parameters (like the run-in-period in our method) we propose in this chapter a new method to deal with the initialisation bias problem. Its main characteristic is the easy and simple way in which it deals with the *initialisation bias* problem, when the method of replications is used for the estimation of steady state parameters. In contrast with the methods discussed in Chapter 4, it does not delete any of the initial observations, it does not need the estimation of any value(s) and it only requires a small modification to the simulation software.

Snell and Schruben, (1979, 1985) report some theoretical results based on the use of different weights of the observations obtained using a simple autoregressive model for the simulated series (instead of the replications method which is used in this chapter). The objective of their study is to analyse the effect of different weighting schemes based on regression techniques. They consider that deleting some of the initial observations is just a special case of observation weighting. This is a theoretical study and as they add, "topics of further study include extension of these results to more complex systems than an AR(1) process, as well as robustness studies of applying these estimators to non-AR(1) processes to see how they fare." (Snell and Schruben, 1985). However, to our knowledge, the method has not been extended for its use on more typical simulation models.

### 6.1.1. CHAPTER OBJECTIVES

We know that the initialisation bias problem is due to the fact that the initial observations recorded in a simulation are usually not representative of the steady state conditions. Although the usual way of dealing with the problem is by deleting some of the initial observations we asked ourselves if this deletion was necessary. The second obvious question was, is it necessary for these initial observations to have the same influence than those obtained for longer simulation run lengths and that are more representative of the steady state values ? We want in this chapter to give an answer to these questions and to do this we will use the well known concept in forecasting techniques of weighted averages. Based on this technique, we want to show that it can be easily modified in order to use it in simulation, and that the mean estimates thus obtained will reach the steady state for shorter simulation run lengths than they would if no attempt is made for dealing with the initialisation bias problem.

Although the method is so simple that it may look obvious, to our knowledge, when the method of replications is used, nobody has suggested it as a way of dealing with the problem of the initialisation bias. As discussed above, the experiment of Snell and Schruben is of a theoretical nature, it uses an autoregressive representation of the data (see Section 5.4.) and no results are given when it is applied to simulation models, especially to complex models using the replications method for the estimation of the steady state parameters.

### 6.1.2. CHAPTER OUTLINE

In Section 6.2 we describe the new method which is based on the concept of weighted averages. Section 6.3 discusses the empirical results used to check the performance of the new proposed method. A further check of the new proposed method is carried out in Section 6.4. using some of the systems discussed in Chapter 4 and for which an analytical answer can be calculated.

## 6.2. USE OF WEIGHTED AVERAGES TO EVALUATE STEADY STATE PARAMETERS.

In Section 6.2.1. we explain the concept of **weighted averages.** In Section 6.2.2. we discuss how to use this concept to evaluate simulation steady state parameters and how its use helps to diminish the influence of the initial conditions.

### 6.2.1. WEIGHTED AVERAGES

"Weighted Averages are useful when the data you are examining are not equally important. You can modify the degree of importance by assigning weighting factors to each value. The weighted average is calculated by multiplying each data value by the appropriate weighting factor, and dividing the total by the sum of the weighting factors. When the weighting factors do not sum one (sic), each weighting factor is multiplied by the appropriate constant to force the sum to equal one." (1989, STATGRAPHICS, Version 4.0)

### 6.2.2. WEIGHTED AVERAGES IN STEADY STATE SIMULATION

By using the concept of weighted averages it is possible to eliminate, or at least greatly reduce, the influence of the initial conditions when we want to estimate a steady state parameter in simulation.

When we obtain an estimate $X_j$ in replication j, this estimate is calculated as the average of N individual observations. To explain this in more detail let us assume that $X_j$ is an estimate of the mean queuing time of queue Q. To obtain this estimate we run the simulation for a period of time R. Every time that we remove a unit from the queue Q, we record the queuing time, $T_i$, (i=1, 2...N) which is the time spent by unit *i* in Q. In this way we obtain N different observations. The queuing time estimate in the *j*th.

replication, $X_j$, is calculated by Equation 6.1.

$$X_j = \frac{\sum_{i=1}^{N} T_i}{N} \tag{6.1}$$

But when we start the simulation, the initial conditions (i.e., queue length of the different queues when are simulating a queuing network) are not representative of the steady state values. This means that some of the first values $T_i$ that we have recorded are quite different from the steady state queuing time and therefore, the value of $X_j$ may greatly differ from the steady state value.

The basic idea behind the WEIGHTED AVERAGES method is then to assign a smaller weight to these initial values so as to minimise their influence on the estimate $X_j$. By doing this the influence of the initial conditions is minimised without increasing too much the standard deviation of the steady state estimate $\bar{X}$, which is obtained as the average of the mean estimates $X_j$. In practice, when we use weighted averages, the estimate $X_j$ obtained in replication *j* is calculated from the simulation as the weighted average of $T_i$ observations (j=1, 2...k) as given by equation (6.2) and where $T_1$ is the first observation that is recorded, $T_2$ the second and so on.

$$X_j = \frac{\sum_{i=1}^{k} w_i * T_i}{\sum_{i=1}^{k} w_i} \tag{6.2}$$

In Equation 6.2. $w_i$ is the weight assigned to observation *i*. In order to eliminate the influence of the initial conditions, $w_1 < w_2 < w_3 ... < w_k$.

One of the characteristics of this method is that the exact value of the weights is not extremely crucial. In other words, in some experiments we used:

$$w_i = i \tag{6.3}$$

while in others we chose exponential values for the weights as shown in Equation 6.4.

$$w_i = (1 - e^{-k2*i}) \qquad (6.4.)$$

with k2=0.0001, and also smaller or larger values. Experiments using different values for k2 were made and this value is not critical in the performance of the new method.

## 6.3. EMPIRICAL RESULTS.

In this section we discuss the results obtained with the method explained in Section 6.2. The simulation models for the experiments and the type of results obtained are described in Section 6.3.1. In Section 6.3.2. (and in Appendix I) we analyse the results for each one of the different simulation models used to test the method proposed in this chapter.

In Section 4.5. we discussed the measures of performance used to compare the behaviour of the run-in-period estimated with the method proposed in Chapter 4 to that of other run-in-periods. Similar measures can be used to define an "acceptable" or "good" performance of the method of the **weighted averages** as compared to the **standard** method for the elimination of the influence of the initial conditions.

**NOTE** : The method usually used in simulation where each individual observation is assigned the same weight will be called "**standard**" throughout the rest of the chapter.

In this context we consider a procedure to be "good" if the mean estimates reach the steady state for shorter simulation run lengths as compared to the **standard** procedure. We also look for mean estimates closer to the steady state value, which means that the difference in absolute value between the mean estimates obtained with the new procedure and the real steady state value $\mu$, is smaller than the same difference for the estimates obtained with the **standard** method.

### 6.3.1. DESCRIPTION OF THE EXPERIMENTS

In order to test the new procedure proposed in this chapter we have used some of the simulation models used in the previous chapters. We analyse in Section 6.3.2. results obtained for the mean queuing time of some of the queues of the PUB model. In Section 6.4. we apply the method to the M/M/1 queue. Results for other simulation models for which no analytical answer exists, as well as for the M/M/4 and the Jackson's system described in Chapter 4, are given in Appendix I.

In some cases we compare the performance of the **weighted averages** method where no initial observations are deleted with the performance of the system when we use a run-in-period. The run-in-period used in these experiments has been evaluated following the procedure proposed in Chapter 4 which as was shown gives good results in most cases.

Our objective is to establish that the influence of the initial conditions is greatly reduced when using the method described in Section 6.2, and that therefore not only should the mean estimates for short simulation run lengths be close to the steady state value, but this state should be reached for shorter simulation run lengths than if all the observations have the same weight or if some of them are deleted. To check that the method of **weighted averages** meets these two conditions, tables with the mean estimates as a function of the simulation run length are given and as in Chapter 4 we compare them to the steady state value which has been approximated in an empirical way in Appendix C. For the simulation models used in this research we have considered that mean estimate values falling within a value $\epsilon=2.5\%$ are sufficiently close to the steady state value and that a smaller value of $\epsilon$ will not make an important difference in practice.

### 6.3.2. ANALYSIS OF THE RESULTS FOR THE PUB MODEL

In this simulation model results for three queues were obtained: WAIT, CLEAN and IDLE. Tables 6.1., 6.2. and 6.3. give the mean queuing time estimates for the CLEAN, IDLE and WAIT queues respectively and for three different methods: the **standard**, the **weighted averages**, and the Run-In-Period method. In these tables we have underlined the values of the mean estimates for which the parameter can be considered to be in the steady state.

The steady state values obtained in Appendix C, as well as the range of values for which each parameter can be considered to be in the steady state are the following:

| **Queue** | **Steady state** | **Range** |
|---|---|---|
| CLEAN | 209.400 | [204.160 , 214.630] |
| WAIT | 1.141 | [1.112 , 1.169] |
| IDLE | 2.001 | [1.950 , 2.050] |

From the underlined values in the tables we can notice that for all the three queues the mean estimates when the new proposed method in this chapter is used will reach the steady state for shorter simulation run lengths than when the **standard** method is used: for the CLEAN queuing time the parameter can be considered to be in the steady state for a simulation run length of 500 when the **weighted averages** method is used, while with the "Standard" method a simulation run length of at least 3000 is required.

For the WAIT queue a simulation run length of 1000 minutes is required for the **weighted averages** estimates to reach the steady state while at least a run length of 7500 minutes is required when the "Standard" method is used.

For the IDLE queue the difference between the simulation run lengths required for the parameter to reach the steady state is similar: a simulation run length of 1000 minutes is required for the parameter to reach the steady state if the **weighted averages** are used, while a simulation run length of at least 7000 minutes is required if the "Standard" method is used.

| CLEAN Mean Queuing Time Estimates | | | | | | |
|---|---|---|---|---|---|---|
| Run Length | "Standard" | | "Weighted" | | Run-In-Period | |
| | Mean | Std.Dev. | Mean | Std.Dev. | Mean | Std.Dev. |
| 500 | 182.299 | 39.137 | 211.882 | 52.420 | | |
| 1000 | 195.886 | 39.217 | 208.511 | 50.759 | 212.953 | 61.036 |
| 1500 | 200.825 | 36.219 | 209.468 | 43.696 | 212.297 | 47.165 |
| 2000 | 203.170 | 32.180 | 209.676 | 38.570 | 211.914 | 39.320 |
| 2500 | 203.921 | 29.802 | 208.619 | 36.744 | 210.660 | 34.747 |
| 3000 | 205.451 | 27.589 | 209.925 | 32.418 | 211.106 | 31.794 |
| 3500 | 205.709 | 25.421 | 209.232 | 29.811 | 210.464 | 28.526 |
| 4000 | 206.445 | 24.195 | 209.843 | 28.894 | 210.631 | 26.816 |
| 4500 | 207.558 | 22.950 | 211.229 | 26.740 | 211.377 | 25.212 |
| 5000 | 208.478 | 21.523 | 212.287 | 24.587 | 211.961 | 23.277 |
| 5500 | 208.760 | 20.397 | 212.149 | 23.626 | 211.911 | 21.993 |
| 6000 | 208.867 | 19.860 | 211.829 | 24.046 | 211.744 | 20.166 |
| 6500 | 208.954 | 18.828 | 211.604 | 22.332 | 211.636 | 20.166 |
| 7000 | 208.700 | 18.042 | 210.711 | 21.176 | 211.115 | 19.074 |
| 7500 | 208.681 | 17.618 | 210.450 | 20.886 | 210.937 | 18.563 |
| 8000 | 208.443 | 17.022 | 209.833 | 20.253 | 210.553 | 17.797 |
| 8500 | 208.461 | 16.207 | 209.726 | 19.291 | 210.441 | 16.863 |
| 9000 | 208.755 | 15.723 | 210.113 | 18.558 | 210.614 | 16.276 |
| 9500 | 208.794 | 15.554 | 210.087 | 18.616 | 210.558 | 16.058 |
| 10000 | 208.920 | 15.329 | 210.196 | 18.310 | 210.599 | 15.800 |
| 10500 | 208.689 | 15.049 | 209.593 | 17.815 | 210.258 | 15.531 |

**Table 6.1.** CLEAN mean queuing time estimates as a function of the simulation run length and of the method of dealing with the initialisation bias problem.

| IDLE Mean Queuing Time Estimates | | | | | | |
|---|---|---|---|---|---|---|
| Run Length | "Standard" | | "Weighted" | | Run-In-Period | |
| | Mean | Std.Dev. | Mean | Std.Dev. | Mean | Std.Dev. |
| 500 | 2.807 | 1.557 | 2.247 | 1.631 | 2.465 | 1.523 |
| 1000 | 2.315 | 1.035 | 2.013 | 1.149 | 2.149 | 1.033 |
| 1500 | 2.218 | 0.829 | 2.040 | 0.884 | 2.108 | 0.828 |
| 2000 | 2.151 | 0.700 | 2.017 | 0.796 | 2.068 | 0.701 |
| 2500 | 2.120 | 0.627 | 2.020 | 0.740 | 2.055 | 0.628 |
| 3000 | 2.105 | 0.572 | 2.025 | 0.646 | 2.050 | 0.572 |
| 3500 | 2.080 | 0.522 | 2.004 | 0.590 | 2.033 | 0.521 |
| 4000 | 2.076 | 0.494 | 2.019 | 0.563 | 2.035 | 0.493 |
| 4500 | 2.074 | 0.463 | 2.028 | 0.519 | 2.038 | 0.462 |
| 5000 | 2.072 | 0.436 | 2.034 | 0.488 | 2.039 | 0.435 |
| 5500 | 2.067 | 0.407 | 2.032 | 0.463 | 2.037 | 0.406 |
| 6000 | 2.059 | 0.391 | 2.024 | 0.462 | 2.031 | 0.391 |
| 6500 | 2.052 | 0.373 | 2.020 | 0.442 | 2.027 | 0.373 |
| 7000 | 2.045 | 0.359 | 2.012 | 0.430 | 2.022 | 0.358 |
| 7500 | 2.048 | 0.350 | 2.022 | 0.418 | 2.026 | 0.349 |
| 8000 | 2.045 | 0.333 | 2.021 | 0.391 | 2.025 | 0.333 |
| 8500 | 2.046 | 0.321 | 2.025 | 0.375 | 2.026 | 0.321 |
| 9000 | 2.039 | 0.311 | 2.015 | 0.359 | 2.021 | 0.311 |
| 9500 | 2.037 | 0.310 | 2.013 | 0.365 | 2.019 | 0.310 |
| 10000 | 2.033 | 0.303 | 2.009 | 0.359 | 2.017 | 0.302 |
| 10500 | 2.026 | 0.293 | 1.997 | 0.346 | 2.010 | 0.293 |
| 11000 | 2.026 | 0.283 | 2.001 | 0.334 | 2.011 | 0.282 |
| 11500 | 2.026 | 0.274 | 2.004 | 0.325 | 2.012 | 0.273 |
| 12000 | 2.021 | 0.267 | 1.995 | 0.317 | 2.007 | 0.266 |
| 12500 | 2.018 | 0.264 | 1.993 | 0.316 | 2.005 | 0.263 |
| 13000 | 2.021 | 0.261 | 2.000 | 0.313 | 2.008 | 0.260 |
| 13500 | 2.021 | 0.254 | 2.002 | 0.306 | 2.009 | 0.253 |
| 14000 | 2.020 | 0.251 | 2.002 | 0.302 | 2.008 | 0.251 |
| 14500 | 2.021 | 0.250 | 2.006 | 0.305 | 2.010 | 0.250 |
| 15000 | 2.020 | 0.247 | 2.005 | 0.302 | 2.009 | 0.247 |

**Table 6.2.** IDLE mean queuing time estimates as a function of the simulation run length and of the method of dealing with the initialisation bias problem.

| WAIT Mean Queuing Time Estimates | | | | | | |
|---|---|---|---|---|---|---|
| Run Length | "Standard" | | "Weighted" | | Run-In-Period | |
| | Mean | Std.Dev. | Mean | Std.Dev. | Mean | Std.Dev. |
| 500 | 0.894 | 0.632 | 0.982 | 0.797 | 0.947 | 0.716 |
| 1000 | 1.018 | 0.778 | 1.112 | 1.057 | 1.052 | 0.847 |
| 1500 | 1.059 | 0.806 | 1.121 | 1.009 | 1.083 | 0.855 |
| 2000 | 1.095 | 0.826 | 1.151 | 1.052 | 1.114 | 0.862 |
| 2500 | 1.110 | 0.786 | 1.150 | 0.949 | 1.124 | 0.816 |
| 3000 | 1.109 | 0.730 | 1.134 | 0.808 | 1.123 | 0.754 |
| 3500 | 1.115 | 0.671 | 1.137 | 0.710 | 1.127 | 0.690 |
| 4000 | 1.103 | 0.602 | 1.113 | 0.580 | 1.114 | 0.617 |
| 4500 | 1.105 | 0.553 | 1.112 | 0.542 | 1.114 | 0.565 |
| 5000 | 1.108 | 0.524 | 1.114 | 0.525 | 1.117 | 0.534 |
| 5500 | 1.106 | 0.492 | 1.110 | 0.493 | 1.113 | 0.502 |
| 6000 | 1.108 | 0.474 | 1.111 | 0.498 | 1.115 | 0.482 |
| 6500 | 1.106 | 0.456 | 1.113 | 0.492 | 1.112 | 0.464 |
| 7000 | 1.110 | 0.443 | 1.118 | 0.492 | 1.116 | 0.450 |
| 7500 | 1.123 | 0.446 | 1.144 | 0.554 | 1.129 | 0.453 |
| 8000 | 1.128 | 0.452 | 1.150 | 0.584 | 1.134 | 0.458 |
| 8500 | 1.128 | 0.437 | 1.146 | 0.550 | 1.133 | 0.442 |
| 9000 | 1.126 | 0.421 | 1.140 | 0.513 | 1.131 | 0.426 |
| 9500 | 1.124 | 0.403 | 1.134 | 0.477 | 1.129 | 0.407 |
| 10000 | 1.125 | 0.397 | 1.134 | 0.482 | 1.129 | 0.401 |
| 10500 | 1.126 | 0.389 | 1.136 | 0.470 | 1.130 | 0.393 |

**Table 6.3.** WAIT mean queuing time estimates as a function of the simulation run length and of the method of dealing with the initialisation bias problem.

In some cases (see tables above and results for the STEELWORKS and the LAUNDERETTE models in Appendix I) the standard deviation of the sample mean when the system is already in the steady state is slightly larger when **weighted averages** are used than when some of the initial observations are deleted (run-in-period) or when equal weights are assigned to each one of the observations (no run-in-period). Nevertheless this slightly larger value of the standard deviation when the **weighted averages** is used is more than compensated by the easiness of the new proposed method and by the fact that no values (like for example the run-in-period) need to be estimated. When a run-in-period is used, this value itself and, in some methods, (see Chapter 4) some parameters, need to be estimated. From the computational and practical point of view the new method is easier to implement and to use. In order to use it we need only to modify some lines in the simulation software. These are the lines where the individual observations are recorded during a simulation run in order to average them at the end of the simulated time. Appendix J gives the PASCAL implementation of the **standard** method and of the modifications that are required to the software in order to use this new method.

## 6.4. WEIGHTED AVERAGES METHOD APPLIED TO A SYSTEM WITH KNOWN ANALYTICAL ANSWER.

In this section we apply the method proposed in this chapter to the M/M/1 model discussed in Chapters 3 and 4. In Appendix I we give results for the M/M/4 and the Jackson's model also discussed in Chapter 4.

Estimates of the mean queuing time and of the mean queue length of the M/M/1 queue with $\tau = \lambda/\mu = 10/15$ are given in Table 6.5. as a function of the simulation run length for the "Standard" method as well as for the new proposed method of **weighted averages**

The steady state mean queuing time is 20, and we consider that the system is in steady state if the mean estimates fall within 2.5% of this value, i.e., if they fall in the range [19.50 , 20.5]. Similarly, the steady state mean queue length is 4/3 and the range of values for which we consider that the system is in steady state is [1.300 , 1.366]. As in the previous tables, we have underlined those values for which the parameter can be considered to be in the steady state.

| M/M/1 Mean Queuing Time estimates | | | Mean Queue Length estimates | |
|---|---|---|---|---|
| Run Length | "Standard" | "Weighted" | "Standard" | "Weighted" |
| 500 | 14.876 | 16.732 | 1.085 | 1.300 |
| 1000 | 17.686 | 19.066 | 1.249 | 1.395 |
| 1500 | 18.335 | 19.197 | 1.279 | 1.373 |
| 2000 | 18.576 | 19.066 | 1.281 | 1.340 |
| 2500 | 18.866 | 19.295 | 1.292 | 1.342 |
| 3000 | 19.047 | 19.401 | 1.297 | 1.337 |
| 3500 | 19.085 | 19.322 | 1.297 | 1.329 |
| 4000 | 19.202 | 19.435 | 1.301 | 1.331 |
| 4500 | 19.350 | 19.623 | 1.309 | 1.341 |
| 5000 | 19.318 | 19.523 | 1.304 | 1.326 |
| 5500 | 19.428 | 19.630 | 1.310 | 1.333 |
| 6000 | 19.459 | 19.636 | 1.311 | 1.333 |
| 6500 | 19.580 | 19.820 | 1.316 | 1.341 |
| 7000 | 19.564 | 19.736 | 1.315 | 1.336 |
| 7500 | 19.624 | 19.820 | 1.318 | 1.339 |
| 8000 | 19.605 | 19.743 | 1.315 | 1.332 |
| 8500 | 19.607 | 19.726 | 1.315 | 1.330 |
| 9000 | 19.676 | 19.842 | 1.319 | 1.337 |

**Table 6.6.** Mean queuing time and mean queue length estimates for the M/M/1 queue as a function of the simulation run length.

From these values we notice that when the **weighted averages** method is used, the mean queuing time parameter reaches the steady state for a simulation run length of 4500, while a simulation run length of 6500 at least, is required for the parameter to reach the steady state when the **standard** method is used. The results are similar for the mean queue length: when the **weighted averages** is used, the parameter reaches the steady state for a simulation run length of 2000. When the **standard** method is used we need a simulation run length of 4000 units of time.

## 6.5. CONCLUSIONS

The method of **weighted averages** proposed in this chapter to evaluate steady state parameters is easy to implement as it does not require the estimation of any value on the part of the simulation practitioner.

We have checked that the method performs well by comparing the mean estimates to the steady state values of the different simulation models used to test the procedure. We expect, as discussed in Chapter 4, and also in Section 6.3. that the steady state will be reached for shorter simulation run lengths than if the **standard** procedure is used. The steady state values have been obtained from empirical results as reported in Appendix C. From the comparison with these values we notice that the **weighted averages** mean estimates reach the steady state for shorter simulation run lengths than the **standard** mean estimates. The experiments of this chapter (see Appendix I as well for other examples) show that in all cases, except when the approach to the steady state is oscillatory, the mean estimates obtained with the proposed method will reach the steady state for shorter simulation run lengths than if the **standard** method is used. There are two possible problems with the **weighted averages** method:

1. There is an increase in the value of the standard deviation as compared to the standard deviation when a run-in-period is used. However, considering that if a run-in-period is used some time has to be spent in estimating its value, we can use this time, that is saved when using the method proposed in this chapter, and increase the simulation run length and the number of replications. This will reduce the standard deviation.

2. Sometimes modification to the simulation software is not possible or they are not easy to carry out. In this case, the only possible solution is to

estimate the run-in-period with the method proposed in Chapter 4 (or any other method, but as we showed, our method is easy and simple to apply and works well for different types of models).

# CHAPTER 7 : CONCLUSIONS

## 7.1. SUMMARY

We have dealt in this thesis with some common problems in simulation for which there is still no satisfactory answer.

In Chapter 2 we set the appropriate scenario required for the discussion that followed. Considering that one of the main aspects of the research reported in this thesis is that it is mainly empirical, we also pointed out in Chapter 2 some important points that the simulation practitioner should consider both before and after running the simulation. The discussion on possible critical queues gives some guidelines to help the practitioner on the identification of the queues that may never reach a steady state, and also of those queues that may require a long simulation run length to reach the steady state.

In Chapter 3 we addressed a question for which there is no satisfactory answer: that of the number of replications to be used in a simulation. We showed how, when the number of replications is small, the random number streams can influence the mean estimates obtained from the simulation. We also showed in an empirical way, that the number of replications has some influence on the estimation of the simulated time for which the curve of the mean estimates becomes horizontal. An increase in this number makes it easier to identify this point in time. Obviously, due to the transient period, the simulation run length required for a parameter to reach the steady state will never become zero; therefore, there exists a number $k$ of replications such that taking more replications will not make a significant difference in the values of the estimates and in the simulation run length required to reach the steady state. In Chapter 3 we gave a simple method that can be used for the estimation of this value $k$ of replications to be used in the simulation of a particular parameter.

In Chapters 4 and 6 we dealt with the problem of the initialisation bias.

In Chapter 4 we suggest a method that can be used for the estimation of a run-in-period. The observations recorded during this period are then discarded, and only observations for simulation run lengths longer than the run-in-period are used for the estimation of the parameter of interest. We showed how the proposed method is simple and easy to understand, and to use. It is not time-consuming and at the same time it does not require any additional programming or complex modifications to the existing simulation software. In Chapter 6 we introduce a new method for dealing with the initialisation bias problem. It is new in the sense that it does not delete or discard any of the observations obtained from the simulations but assigns different "weights" to these observations. In this way by assigning smaller weights to those observations recorded for short simulation run lengths and that are not significant of the steady state values we are reducing their influence on the mean estimate obtained from the simulation.

In Chapter 5 we discussed the batch means method as a way of estimating steady state parameters. The batch means method as an alternative method for the estimation of steady state parameters works better when DESCRIPTIVE SAMPLING (D.S.) is used. It was also shown that some of the proposed procedures for the use of the batch means method do not work well in practice. For example, we showed that the number of batches does not make a practical difference in the value of the mean estimate or in the width of the c.i. At the same time, as the c.i. relative precision is shorter when D.S. is used we will require a shorter simulation run length to obtain an accurate estimate if we use D.S. than if we use R.S. The reason why some of the procedures proposed in the literature to be used with the batch means method do not work well in practice is that they have not been tested for complex simulation models.

## 7.2. CONCLUSIONS

We may summarise the main general contributions of this thesis to the area of simulation as follows:

1. We show that there is not always the need in simulation for complex, sophisticated, and time consuming procedures. Some problems can be dealt with simple to use and easy to understand procedures. If we compare the methods proposed in this thesis with those that have been proposed in the literature, especially those supported by a mathematical theory, ours are simple and do not require special modifications to the simulation software. This is important because until a method is found to be so useful that it will be incorporated in all the simulation software packages, or at least in most of them, modifications to the software require time and even programming experience. It is true that the proposed methods do not provide the "optimal" answer but, on the other hand, it is not known if a given problem has an optimal answer and at times it is not even worth to look for an "optimal" answer if the one that already exists is sufficiently good.

2. Proposed procedures and techniques in simulation should be tested on complex models. The use of simple systems, for which an analytical answer exists, may give misleading results. The reason for this is the lack of interaction among elements of the simulation, interaction that exists in complex models. This point is proved specifically for the batch means method where the suggested stopping criterion is related to the confidence interval half-width. Unfortunately this value is not only model but parameter dependent.

In a more detailed level, the following are the main contribution of this thesis to the area of simulation:

1. We have identified some aspects concerning the influence of the number of replications on the mean estimates obtained from the simulation that to our knowledge have not been mentioned in the literature before. Therefore, we have been able to formulate a simple method for the estimation of the number of replications to be used in the simulation of a particular parameter.

2. Most of the methods proposed in the literature in the 1980's have a strong mathematical support, and therefore, they are either too complex to use or they require complex programming in order to use them. However, the

problem of the initialisation bias is only a small part of the simulation. It can be avoided by simply increasing sufficiently the simulation run length. For this reason the time spent in dealing with the initialisation bias problem should be only a small fraction of the total amount of time spent in the solution of the problem of interest. Therefore, if a method is going to be useful it should not be complex, and time consuming and it should not require, if possible many additional calculations. The two methods proposed in this thesis (Chapters 4 and 6) meet these requirements. At the same time we have showed that they work well with different types of models and this proves their generality.

3. We have extended the applications of the new sampling method called Descriptive Sampling (D.S.), and we have shown that it performs particularly well when used in the batch means method. At the same time we have shown why we should avoid, if possible, having to set parameters in order to use a particular procedure, like for example using the confidence interval relative precision as a stopping criterion in sequential simulations. Because D.S. gives mean estimates with smaller standard deviation, the confidence interval relative precision will lie in smaller ranges than those when random sampling (R.S.) is used.

If we are going to summarise in a few lines the important points of this thesis we can say that it is an innovative thesis in the area of simulation. By this we mean that the empirical approach that we have followed is one rarely used in this area. But, as we have showed with several examples for both simple and complex simulation models, it gives good results as it helps the researcher in the identification of facts that are common to different types of models. In this way we have been able to propose procedures that do not require great modifications of the existing simulation software. We are not claiming that it is possible to find simple and easy to use procedures to deal with all the problems of simulation, but that this empirical approach can help sometimes.

## 7.3. FUTURE RESEARCH

As Whitt (1989b) says, " Simulation experiments are like exploring trips. We usually have goals, but the interesting discoveries often come from the unexpected." This means that after the research reported in this thesis several questions remain to be answered. As in most research, each question that we answer brings out several related questions: What if ? , Can this alternative method be used ? , Can we set the initial conditions in a different way ? etc. We describe in the rest of this chapter the main points that we think deserve further research.

### 7.3.1. USE IN VERY COMPLEX SIMULATION MODELS

The different procedures proposed and extended in this research were evaluated not only for simple systems with known analytical answer but also for more typical simulation models. However, even if the simulation models used do not have a known analytical answer, they are still more academic than real-life problems. Therefore, an area of future research is the evaluation of these new procedures with real-life simulation models, hopefully for which some practical data exists. Nevertheless this does not mean that the models here employed are all similar and simple. As a matter of fact, the models used in the research are of a different nature. Some of them are "closed" (i.e., STEELWORKS), while others are "open". In some of them the steady state is reached in an oscillatory way (i.e., MILITARY) while in others this approach is monotonic. In some of them there is "Feedback" to one of the queues which in practical systems may lead to instability and a common thing in these models, as opposed to the more commonly used simple systems by the simulation theoreticians, is that there are many interactions amongst the entities of the system and that at the same time there is random sampling from more than two distributions.

### 7.3.2. DIFFERENT PARAMETERS

This thesis proposes different methods and procedures. All of them were checked for several simulation models but using only two simulation parameters: queuing time and queue length. The next step of this research should be the extension of all these procedures to other parameters: entity utilization factor, total time in the system, etc.

The simulation models used to obtain the empirical results reported in this thesis are all of the queuing networks type. Therefore, further research should be done on the generalisation of the different proposed procedures to other types of simulation approaches.

### 7.3.3. IMPLEMENTATION OF DESCRIPTIVE SAMPLING

It is clearly shown, as said in Section 7.2, that D.S. performs better than R.S. when the batch means method is used for the estimation of steady state parameters. Therefore, the next step in the research related with this topic should be the implementation of D.S. in different types of simulation software. This has already been done for the simulation system VS6, used in this research (See Saliby and Paul, 1992). However, it would be interesting to extend this implementation to some of the other popular simulation languages, like SIMSCRIPT II.5, GPSS, etc.

### 7.3.4. SOFTWARE IMPLEMENTATION

For the methods proposed here to be of some use the existing simulation packages should be easily modified to include them. Some work has been done (not reported in this thesis), modifying the VS6 simulation software to automate it. In that way, terminating simulation can be run until some c.i. relative precision is obtained, and for steady state simulations, in a completely automatic way, it has been possible to estimate steady state parameters using either a run-in-period or weighted averages for dealing with the initialisation bias problem. Estimation of the run-in-period does not require a spreadsheet, as Pascal (and some other high-level languages as well) has excellent graphics management. However, further research on the interface of this modification to VS6 is necessary since the package is meant to be used by people with no special knowledge on the topic.

### 7.3.5. ALTERNATIVE STATISTICS: BAYESIAN AND NON-PARAMETRIC

In general we found that in terminating simulations the individual observations will very likely contain several outliers. How serious this problem is depends on the system itself. But in these cases, the use of the sample mean has been seriously questioned. Alternative estimates should be considered, like the MEDIAN for example. It should also be interesting to do more research in the use of non-parametric statistics in these cases (Efron, 1981; Fraser, 1957; Withers, 1983). The reason for this is that they do not require the classical assumptions (especially that of normality) to be met in order to be valid. Further research would also be useful in the area of trimmed means. In the case of trimmed means we simply delete the more extreme observations and because of the likely presence of outliers in the output of terminating simulation this alternative method of analysis should be further studied. Nevertheless this requires a careful and detailed study as in recent research we found that sometimes what would be called outliers from an statistical point of view, are not so. Without these values the standard deviation and the mean estimates will be biased.

Another possible alternative to the classical statistical methods for the analysis of simulation output is the use of Bayesian Methodology. Some research has been reported in this area (Andrews and Schriber, 1983), but it still requires more research.

### 7.3.6. SIMULATION RUN LENGTH

One of the questions that requires more investigation is on how long the simulation run length should be if we want to estimate a steady state parameter. That was not taken into account in this research and the system was considered to be in the steady state when the mean values seem to converge to a specific value. Usually this detection has been done with the help of a line graph. Although this problem is addressed by Whitt (1989a), he applies his method to simple queuing models where the arrival rate to each queue is known. Whitt's approach would not be useful for more complicated systems where the arrival rate depends on the interaction of different entities and their "fight" for limited resources. A more promising approach to the problem of how long should the simulation be was proposed by Duersch and Schruben, (1985) using standardised time series. However, as with most procedures results are reported for some simple simulation models: M/M/1 queue, an (s,S) inventory cost model, and a uniform random number model.

### 7.3.7. PROBLEM OF INITIALISATION BIAS: ALTERNATIVES.

In the problem of the initialisation bias several other alternatives can be considered like the use of initial conditions more similar to those of the steady state. Some research not reported in this thesis was done in this respect and it seems apparently that with a single replication it is possible to define the range where the steady state values lie and therefore this information can be used to set initial conditions more similar to those in the steady state. In this respect, Kelton and Law (1985) report that "the optimal state for initialization tends to be larger than the mean (which in turn exceeds the mode), so that a rough rule of thumb might be to obtain an a priori estimate of the mean number in the system, and then initialize with at least that level of congestion. While such a rule certainly does not guarantee optimal initialization, it should prove better than empty and idle for many models."

When using the batch means method it has been suggested in the literature that some initial observations should be deleted in order to eliminate the initialisation bias problem. However, no method is suggested in this respect. Some experiments not reported in this thesis were carried out, and apparently there is not a great improvement when some initial observations are deleted because the batch size should be large enough to avoid correlation and therefore, the batch mean for the first batch is not greatly influenced by the initial conditions. Nevertheless, some research is required in this respect. Because the batch size should be large enough for the batch means to be uncorrelated, it may not be necessary to delete any observations at all.

### 7.3.8. PROPORTIONS RATHER THAN MEAN ESTIMATES.

Although in this research only sample means were used, in Chapter 5 it is pointed out that if it is possible to formulate the problem in terms of proportions they may give a more accurate and reliable estimate (an extensive discussion on ratio estimation is given in Fieller (1954) and Tin (1965)). This should be another point of future research, especially suited for the use of **descriptive sampling**.

### 7.3.9. OPTIMISATION

This is a field particularly suited for **descriptive sampling** because of the smaller variance of its estimates as most of the proposed procedures in the area of optimisation require small variance of the estimates. "Simulation is commonly used to find the best values for decision variables for problems which defy analytical solutions. This objective is similar to that of optimisation problems, and thus, mathematical programming techniques may be applied to simulation. However, the application of mathematical programming techniques to simulation is compounded by the random nature of simulation responses." (Safizadeh, 1990). Some research has been done in this area of optimisation and one of the most promising methods is based on *Response Surface Methodology* (Biles, 1975; Box and Draper, 1959; Brightman, 1978;

Carrol, 1961; Cooley and Houck, 1982; Schruben and Cogliano, 1985; other references not mentioned here are given in the bibliography). When we are interested in single-response optimisation "the random nature of simulation observations makes the statistical properties of response surface designs appealing to simulation users" (Safizadeh, 1990) as long as a variance reduction technique can be adopted. Therefore, due to the great reduction in variance when using **descriptive sampling** the field of optimisation seems tailor-made for its use. A good survey of the optimisation-by-simulation state of the art can be found in Safizadeh (1990) and Schruben and Jacobson (1989).

## 7.4. FINAL WORDS

The material included in this thesis seems incredible simple, especially if we compare it to procedures that have been proposed in the literature to deal with the same problems. Nevertheless, "we cannot propose complex solutions where simple ones would be adequate. Doing this would ruin the reputation of science, and rightly so. We would not be much better than the car repairman who rebuilds the engine when a tune-up would do. Moreover, many simple ideas are the result of extensive research, often including many wrong starts and detours. What seems to be simple in the end is often the product of hard work. There may be the occasional child who discovers that the emperor has no clothes, or similar facts of great importance, but these are exceptions rather than the rule. The art is to see where simple solutions are adequate. Moreover, even in cases where the underlying phenomena are quite complex, the essential results can often be extracted in such a way that everyone can use them. The discovery of the normal distribution, for instance, was definitely nontrivial. Yet, most first-year students have no problem working with it.

The world is complicated, but this does not mean that we have to make it even more complicated by rejecting simple solutions as trivial. To the contrary, we should strive for simplicity, and we should even try to explain complex relations in a simple way." (Grassman, 1986).

# APPENDICES

# APPENDIX A : ACTIVITY CYCLE DIAGRAMS

## A.1. INTRODUCTION

The use of simulation to analyse the behaviour of a system requires "the setting up of a model of the system under study, in which all relevant components are defined, and the way in which they change through time and affect each other are exactly defined." (Paul and Balmer, 1985). The type of model depends in some way on the type of software that will be used to carry out the simulation, as well as on the questions that the simulation is going to answer.

The elements of a system will interact through time and this interaction should be clearly and carefully described by the model that is used to define the system. However, most real-world systems are very complex and as a first approach, when describing the system in order to obtain a model, we should not try to show all the complexities of the system. In other words, in order to obtain a good model of the system it is necessary to understand its basic behaviour and, to begin with, we have to describe only those relevant parts of the system and how they interact with each other without explaining in detail each one of its complexities. This understanding can be obtained with the use of an ACTIVITY CYCLE DIAGRAM. Examples of real systems that have been modelled using this technique can be found in Chew et al, (1985); El Sheik et al, (1985); Williams et al, (1989); Holder and Gittins, (1989) among others.

**ACTIVITY CYCLE DIAGRAMS** are "one way of modelling the interactions of system objects and are particularly useful for a strong queuing structure." (Paul and Balmer, 1985). An Activity Cycle Diagram shows, in a graphical manner the way the different entities of the system interact with each other. Using an Activity Cycle Diagram we may describe the different states of a system. To draw an Activity Cycle Diagram we use two different symbols. **A circle** represents the periods of time when an entity is "IDLE" **(dead state),**

or waiting to start an activity. In this case we say that the entity is in a **queue.** A **rectangle** represents an activity (**active state**). By using these two symbols it is possible to show the "life story of each class of entity" (Pidd, 1992) and to describe how the different classes of entities interact with each other. However, it is important to bear in mind that different types of entities will in general be engaged in different types of activities. If an Activity Cycle Diagram is going to represent this in a clear way, different "paths" or cycles should be drawn for different types of entities. This means that "each class of entity is considered to have a life cycle which consists of a series of states. The entities move from state to state as their life proceeds." (Pidd, 1992). For the sake of clarity of the diagram, the following restriction is imposed: in the life cycle of any entity, the dead and active states must always alternate. This means that when drawing an Activity Cycle Diagram there will always be a queue (if necessary, it will be a *dummy queue*) between two activities. This applies to each life cycle.

An Activity Cycle Diagram while very clear has many limitations as it does not represent in an exact way all the different characteristics of the real-world system. Nevertheless this is not a disadvantage. If the system that is being simulated is very complex, trying to deal with all the complexities of the system from the very beginning is a potential source of errors and the simulation practitioner will be easily confused and lost in what he is doing. On the other hand the Activity Cycle Diagram, by giving a somewhat simplified vision of the system, will reduce this possible source of errors.

Once an Activity Cycle Diagram has been drawn, it is very easy to obtain a simulation code for the model using a program generator such as VS6. (Paul, (1988); Crookes et al, (1986); Paul and Chew, (1986); Balmer and Paul, (1986)). This particular program generator VS6 is used throughout this research.

## A.2. SOME EXAMPLES

This section describes some systems that can be modelled by an Activity

Cycle Diagram (A.C.D.). These examples, as well as some additional systems that can also be described by an A.C.D. are used to obtain the empirical results that are analysed in this thesis.

### A.2.1. The PUB

This is a typical example used in simulation "since its background is implicitly understood by most readers." (Balmer and Paul, 1985). To explain how an Activity Cycle Diagram is drawn, we will start with an oversimplified version of a PUB. In this version there are just three entities: costumers, glasses, and barmaids. The customer **WAITS** until both a glass and a barmaid are available to **POUR** a drink for him and then, he **DRINKS** it. Similarly, the barmaid is either **IDLE** or **POURING** a drink. The glass is **WAITING** to be used (EMPTY), is **POURED INTO**, is **FULL**, or is **DRUNK FROM**. From this definition of the problem it is possible to identify the different life cycles for each class of entity. This is described in the following table:

| ENTITY | ACTIVE STATES | DEAD STATES |
|---|---|---|
| Customer | Drink | Waits |
| Barmaid | Pour | Idle |
| Glass | Drink From<br>Pour into | Empty<br>Full |

As there are three different classes of entities, there should be three life cycles in the A.C.D. (Activity Cycle Diagram), which are shown in Figure A.1. Combining these three life cycles we obtain the A.C.D. shown in Figure A.2.

We have identified with this simple example the basic steps that should be followed in order to draw an A.C.D.; it is now possible to obtain the A.C.D. of a more realistic version of the PUB.

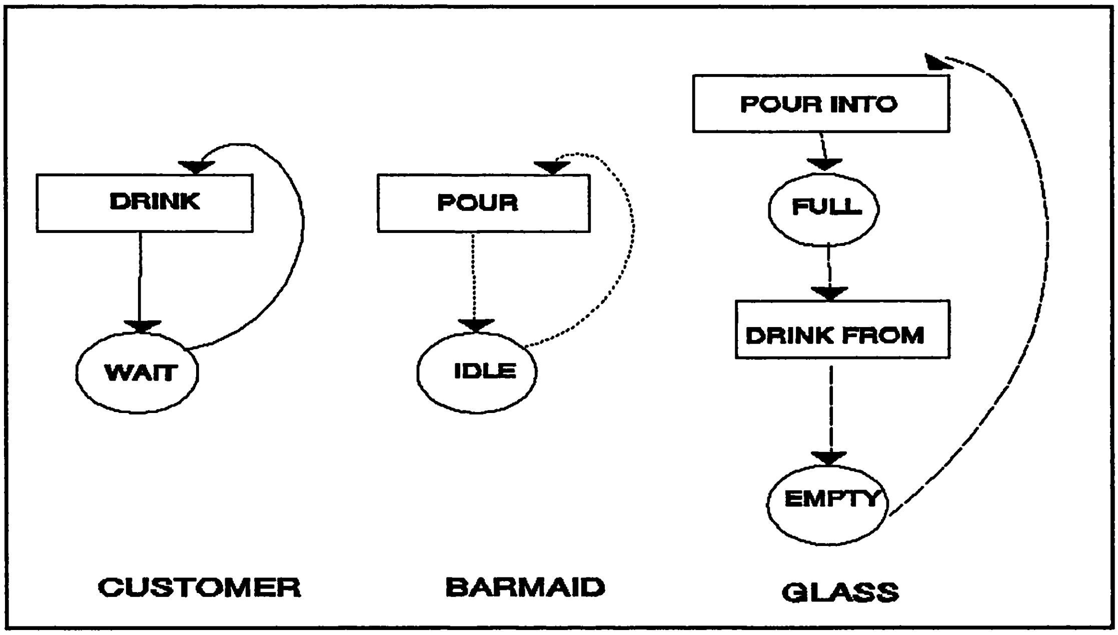



Figure A.1. Cycles for the three different entities in a simple PUB simulation model.

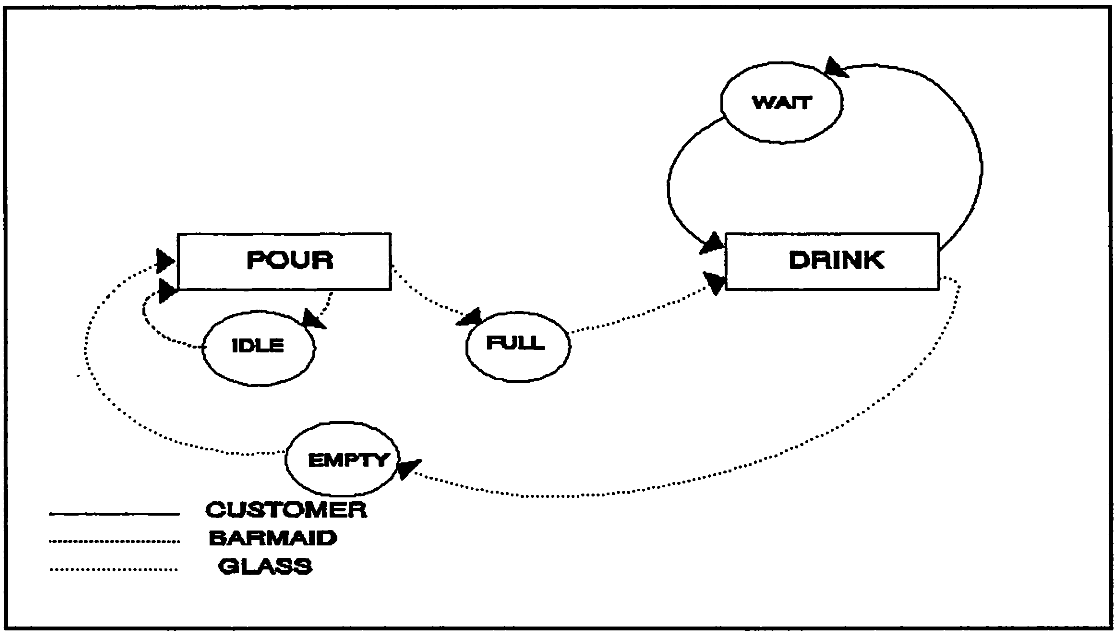



Figure A.2. A.C.D. of a basic simulation model for the PUB system.

In this new version the customers do not "live" in the pub (notice in the above A.C.D. that the customers are always present in the system) but they **ARRIVE** in a random fashion. From statistical theory, this random arrival can be modelled using a negative exponential probability distribution. They **WAIT** until a glass and a barmaid are available so that a drink can be **POURED.** However in real life some customers will have more drinks than others. To model this an **ATTRIBUTE** is assigned to each customer upon his arrival. This is modelled by sampling from the appropriate probability distribution. To identify it, unless some data is available, it may be necessary for the simulation practitioner to obtain data directly at the pub and to use goodness-of-fit tests to determine which probability distribution describes better the pattern of drinks of customers. (Several articles have been published concerning the problem of identifying a suitable input distribution or in other words, of modelling input processes in simulation experiments. Further information can be found in DeBrota et al, 1989; Avramidis and Wilson, 1989; DeBrota et al, 1988; Cochran and Cheng, 1990.)

Once a customer has been served a drink, he will **DRINK** it. But this is the simulation of a real-world problem which means that, at least in theory, before a used and therefore dirty glass can be used again to be poured into, it should be washed. Only when the barmaid is not engaged pouring drinks can she **WASH** those glasses that are dirty. When the customer has finished his drink he will decide if he will have another drink; in simulation this is decided according to the value of the attribute that has been assigned to the customer upon his entrance to the pub. If he will have another drink he will **WAIT** again for a barmaid and a glass to be available. Otherwise he will leave the pub.

From this description of the problem we show in Table A.1. the different states for each type of entity.

Figure A.3. shows the life cycles for each one of the three different classes of entities. Combining these independent life cycles we obtain the A.C.D. shown in Figure A.4.

| ENTITY | ACTIVE STATES | DEAD STATES |
|---|---|---|
| Customer | Arrive<br>Drink<br>Leave | wait for drink<br>ready to drink |
| Barmaid | Pour drink<br>Wash glass | Idle |
| Glass | Pour Into<br>Drink From<br>Washed | Full<br>Dirty<br>clean |

**Table A.1.** States of each one of the three entities of the complete PUB model.

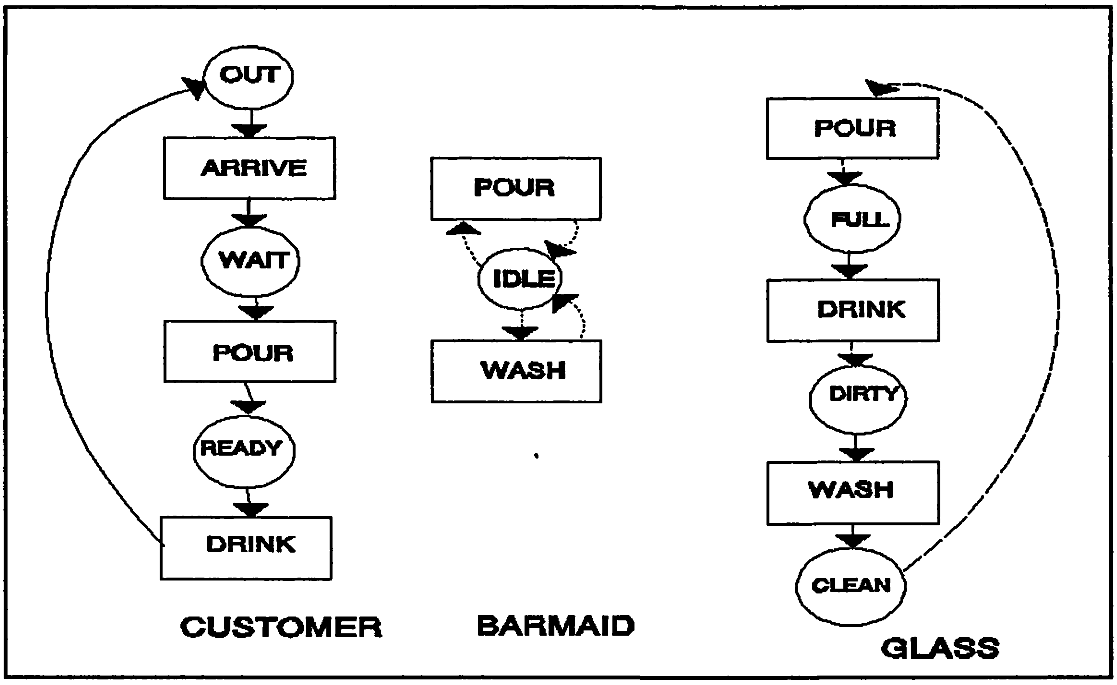



**Figure A.3.** Cycles for the different entities of a complete model for the PUB system.

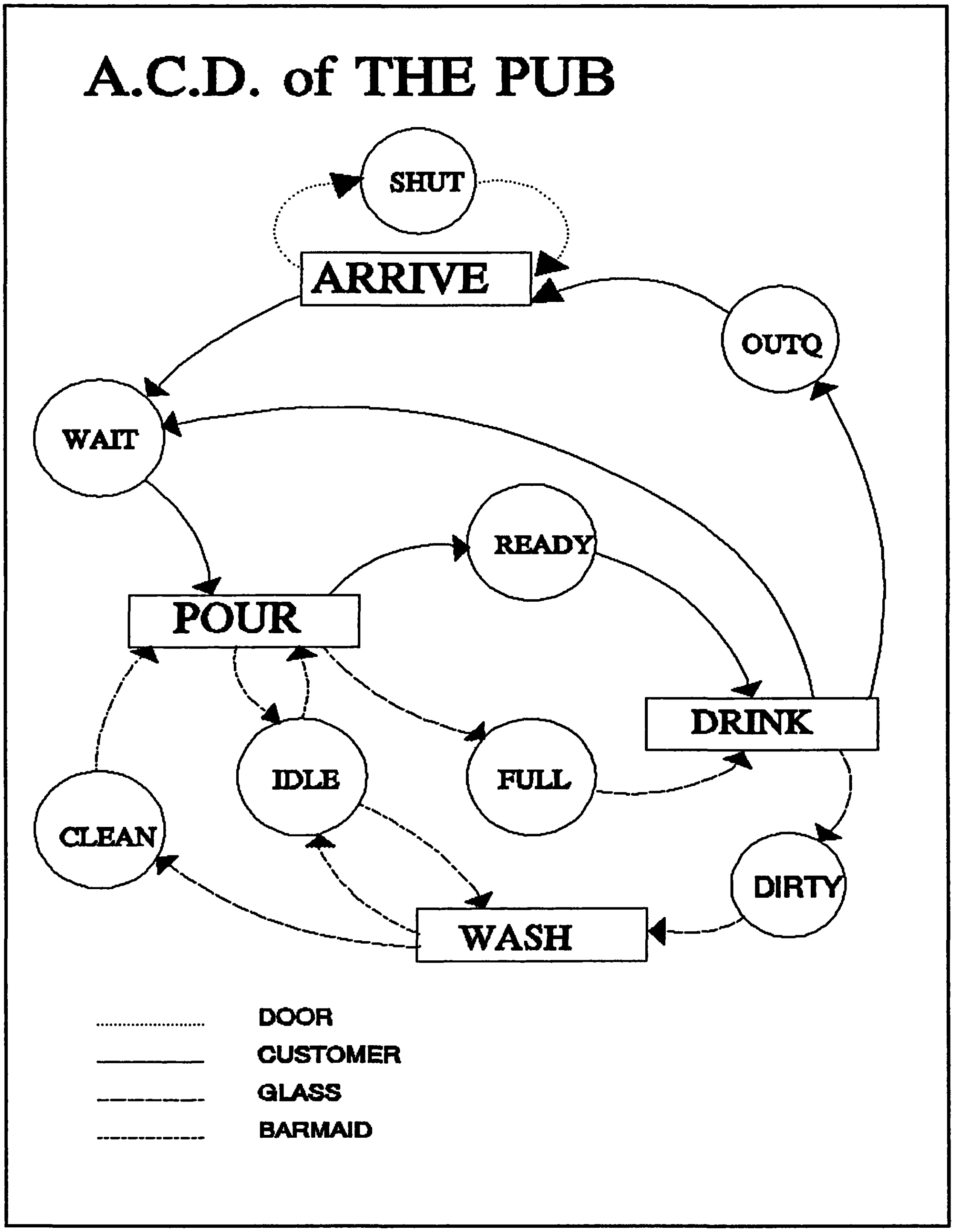



Figure A.4. ACD of a simple model for the PUB system.

### A.2.2. The LAUNDERETTE Example

"Customers arrive in a launderette and queue for one of N1 washing machines (no capacity problems!). On completion of washing, the customer puts the washing into a basket (if one of the N2 baskets is available) and carries the washing to a drier. There are N3 driers." (Balmer and Paul, 1985)

The A.C.D is shown in Figure A.5.

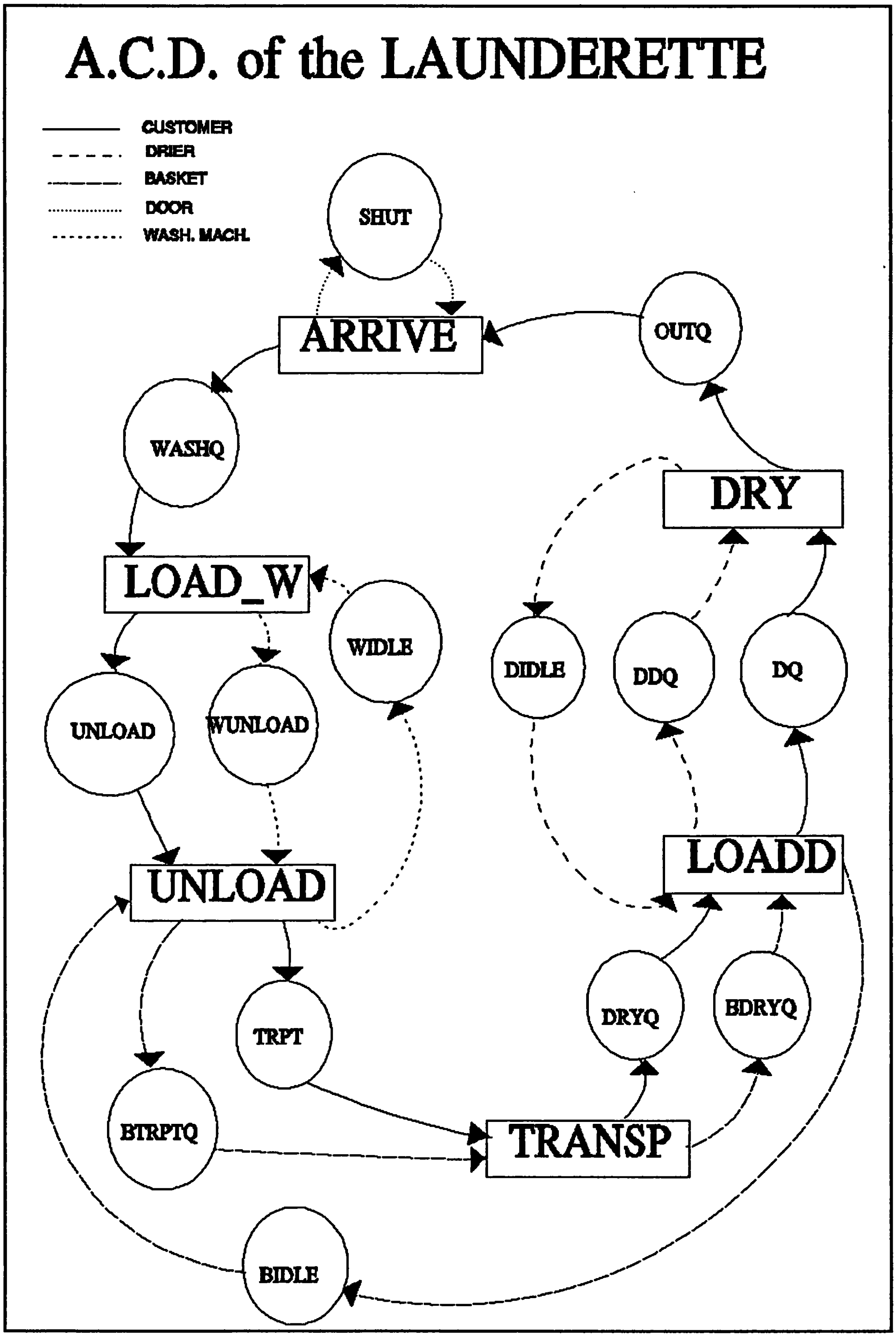



**Figure A.5.** A.C.D. of the LAUNDERETTE simulation model.

### A.2.3. The STEELWORKS example.

This simulation problem deals with the operation of a steelworks. A layout of the proposed steelworks is illustrated in Figure A.6 and the A.C.D. is given in Figure A.7.

"The functional characteristics are as follows:

a. **Blast Furnaces** A "cast" is the amount of molten iron that the furnace melts in one go and is a stochastic variable as far as this simulation is concerned. The time a blast furnace takes to process the iron ore is also a stochastic variable.

b. **Steel Furnaces**

Each of the five steel furnaces takes exactly 100 tonnes of molten iron per charge. The time a steel furnace takes to process the iron ore is also a stochastic variable.

c. **Crane**

The crane travels along an overhead gantry and carries a ladle (a large spoon shaped vessel) which holds 100 tonnes of molten iron. The crane is filled at the pit from one torpedo at a time. The pit is 2-sided to avoid a delay if the crane requires more than one torpedo to load it. In the event that two torpedoes are insufficient to load the crane, the time taken to discharge the second torpedo can be considered sufficient for a third torpedo (if there is one) to take the place of the first, ready for unloading. Note: The pit's function is to catch any spillage of molten iron. It does not hold molten iron.

d. **Torpedoes**

Each torpedo can hold up to 300 tonnes of molten iron. When a blast furnace has emptied its blast into the minimum number of torpedoes required (if available) all torpedoes with molten iron go to the pit (by railway track) This includes partially full torpedoes.

e. **The problem**

If a torpedo is not available to catch the molten iron when a blast furnace blows, the molten iron is dropped on the floor (otherwise the furnace would be ruined). (Balmer and Paul, 1985).

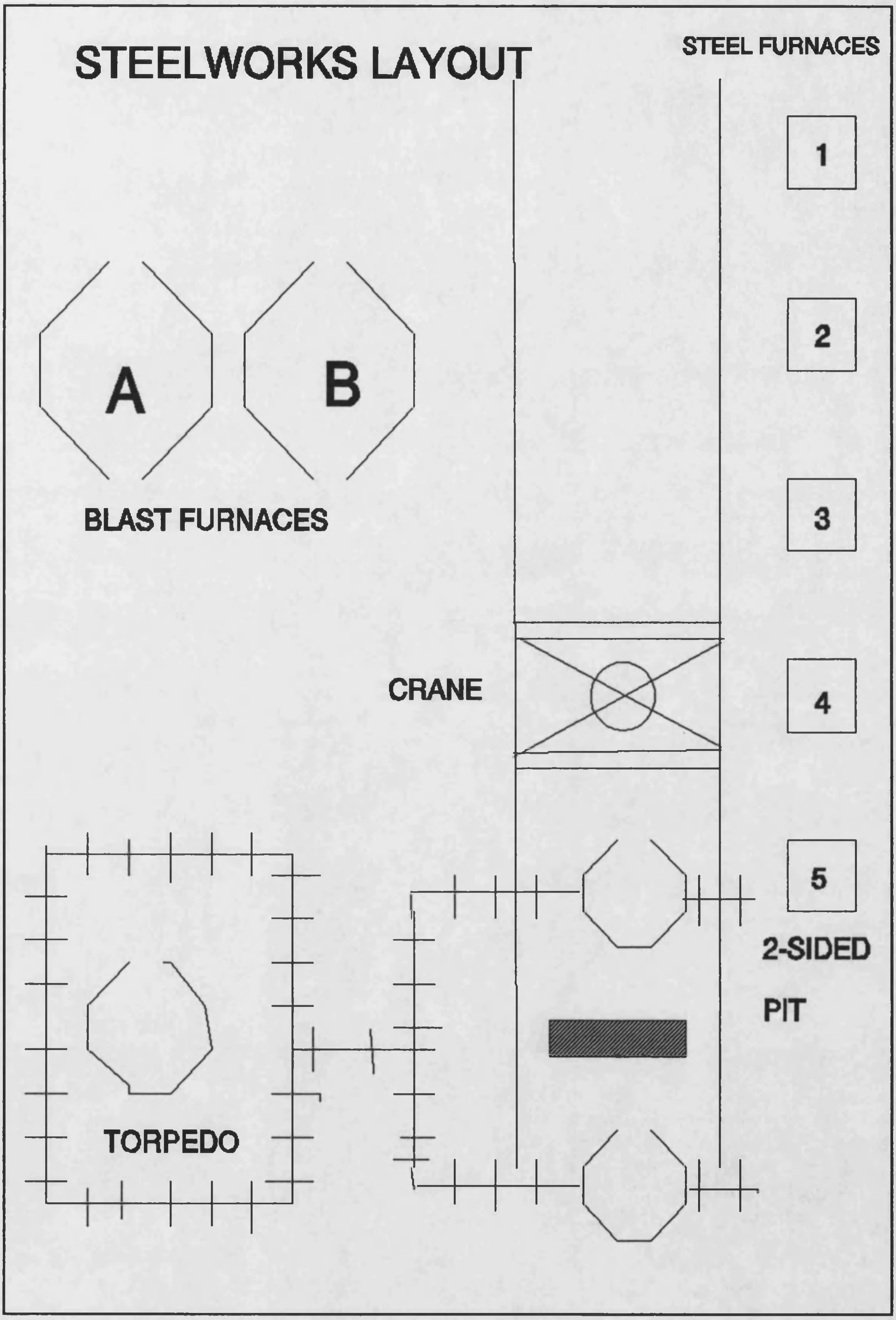



Figure A.6. STEELWORKS layout.

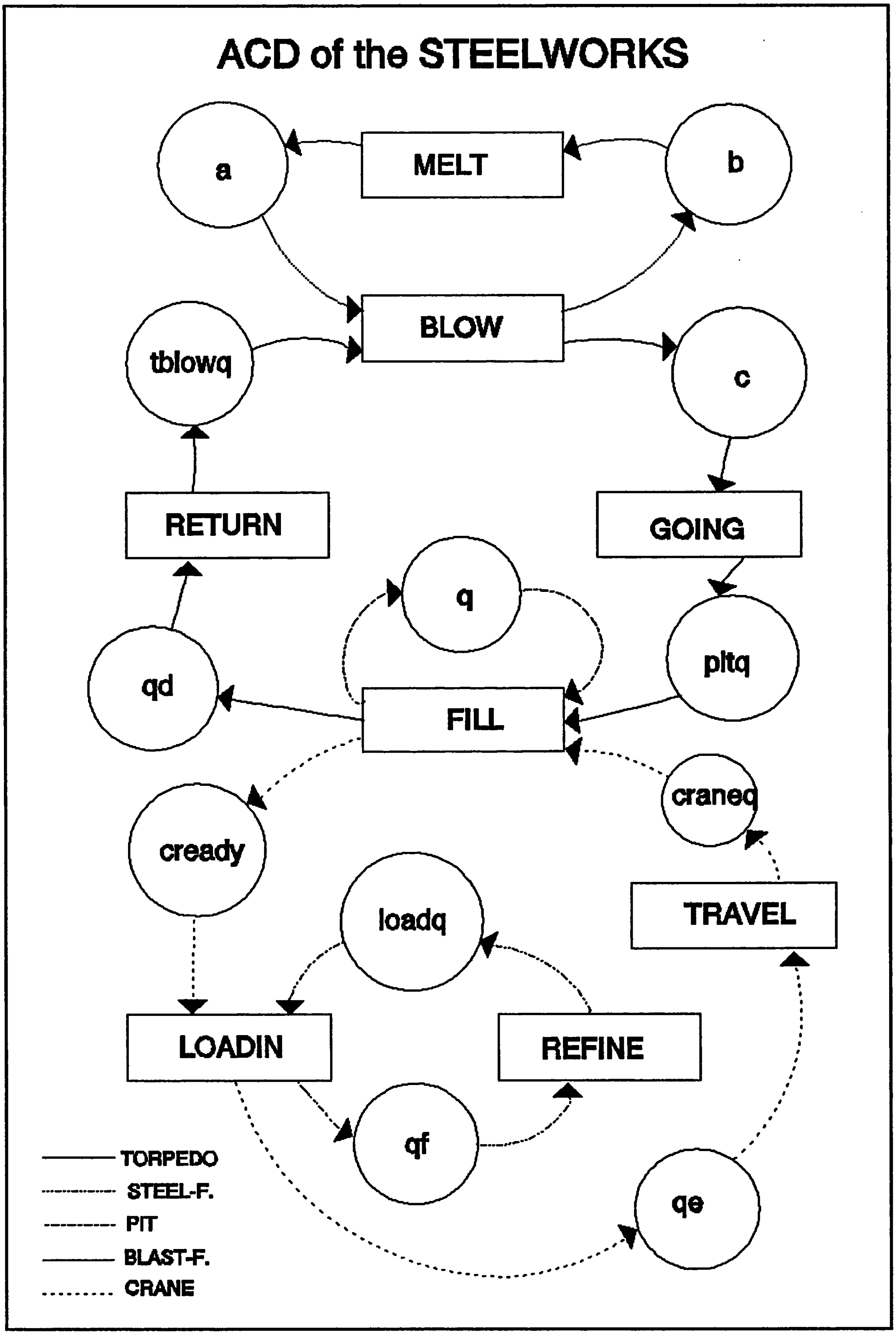



**Figure A.7.** A.C.D. of the STEELWORKS simulation model.

### A.2.4. THE FISH PACKING Example

"Fillets arrive randomly with a mean arrival time of 5 minutes at a packing station where each one is weighed. The weight of the fish fillets are sampled from a normal distribution between 280 - 340 g and the weighing time is 2 minutes. After weighing, the fish fillet is packed into one of the available boxes that has sufficient capacity, otherwise it is recycled (boxes have a capacity of 4000 g). The packing time is normally distributed with a mean of 5 minutes and a variance of 1 minute. When a box has reached a certain minimum capacity, such as 3700g, it is replaced by an empty box. A micro-computer is connected to the scales and instructs a man or mechanism about the allocation. Thus the computer knows at all times the current weights of the boxes. The problem is to devise a good algorithm which minimises the underweight of the boxes." (Chew, 1986). The A.C.D. is shown in Figure A.8.

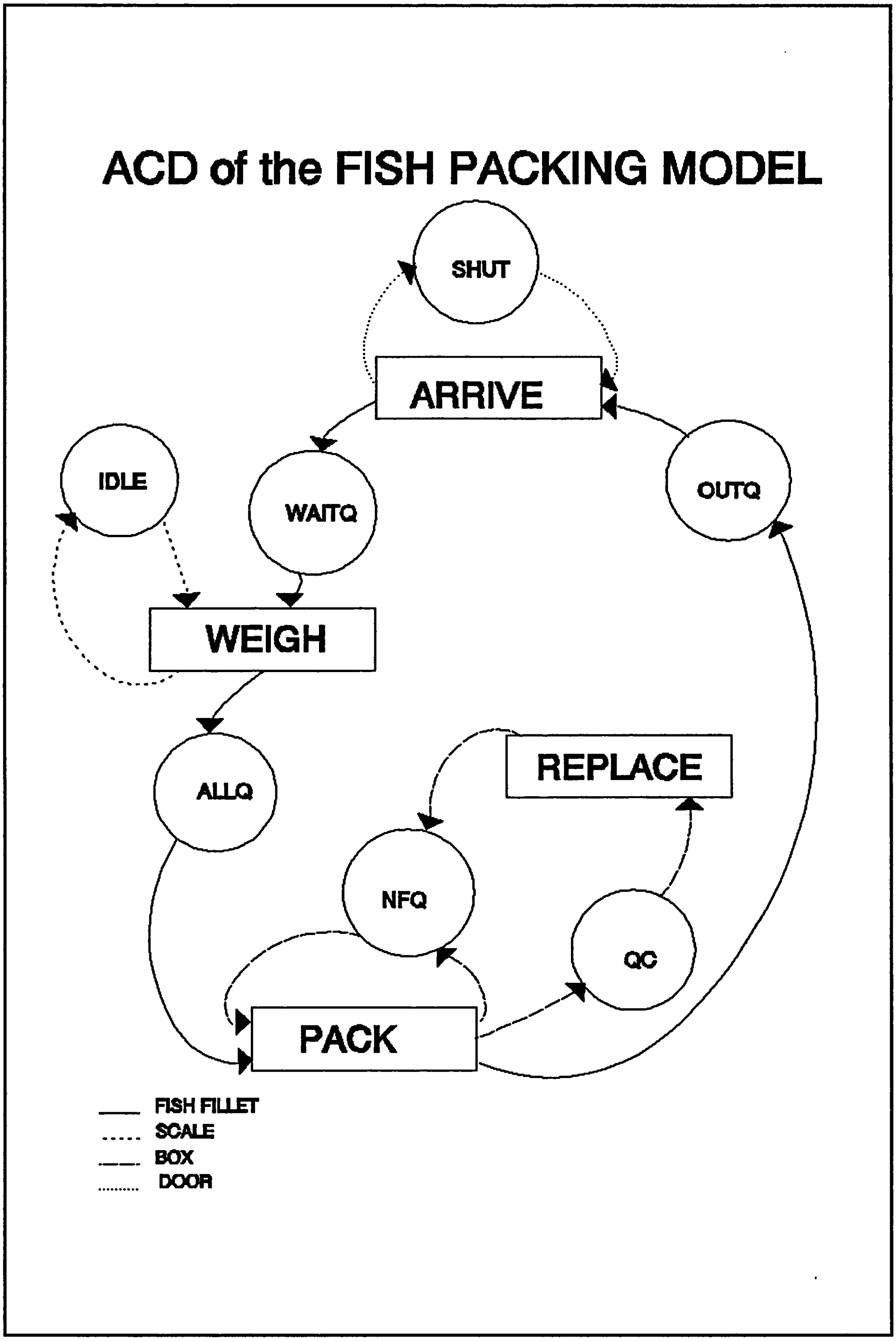



Figure A.8. A.C.D. of the FISH PACKING model.

## A.2.5. THE BRAZILIAN HOSPITAL Example

In this model there are four entity types: patient, doctor, receptionist and consultant as shown in Figure A.9.

"Patients arrive in the system every two minutes. They queue for the receptionist which takes 10 minutes for reception and then queue for the doctor which takes Negexp(2) for the consultation. Upon completion of consultation, 50 percent of the patients go to the outside queue; 30 percent of the patients go to the queue out as out-patients and queue for one of the 10 consultants which take Normal(40,1) minutes. The remaining patients go to the entry queue for the hospital. The entry time to the hospital is Negexp(6) minutes and the patients stay in the hospital for Normal(72*60,24*60) minutes.

The doctor is idle unless engaged in consultation. The receptionist is idle unless engaged in reception, enter or outpatient. Activity outpatient has a higher priority than the other two activities. The consultant is idle unless engaged in doctor." (Chew, 1986)

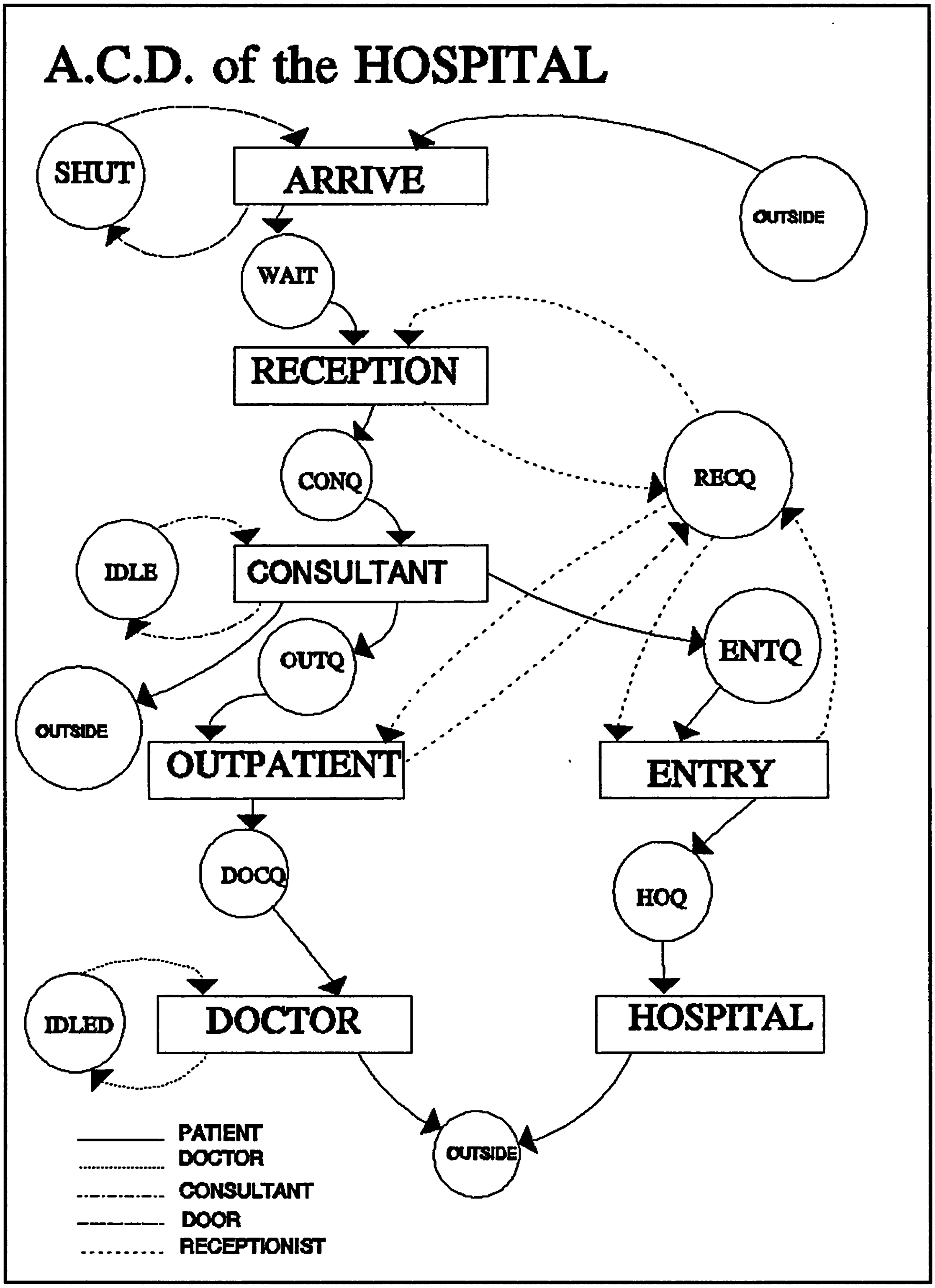



Figure A.9. A.C.D. of the BRAZILIAN HOSPITAL model.

### A.2.6. THE MILITARY Example

In this military model, the entity types are day, crew, helicopter, target and door, as shown in Figure A.10.

"Targets arrive randomly with a mean arrival time of 20 minutes. They queue for one of the 3 helicopters and one of the 3 crews which take Normal(30,5) to intercept. If none of the helicopters or crews is available, the targets take 1 minute to be missed.

The helicopter is on the ground unless engaged in intercept. For the crews, they are idle unless engaged in sleep or intercept. Sleeping time is 8 hours or 480 minutes. The days awake into the queue moon for the activity sleep. After sleeping, the days go to the queue sun and back to the activity awake. Awaking time is 16 hours or 960 minutes." (Chew, 1986).

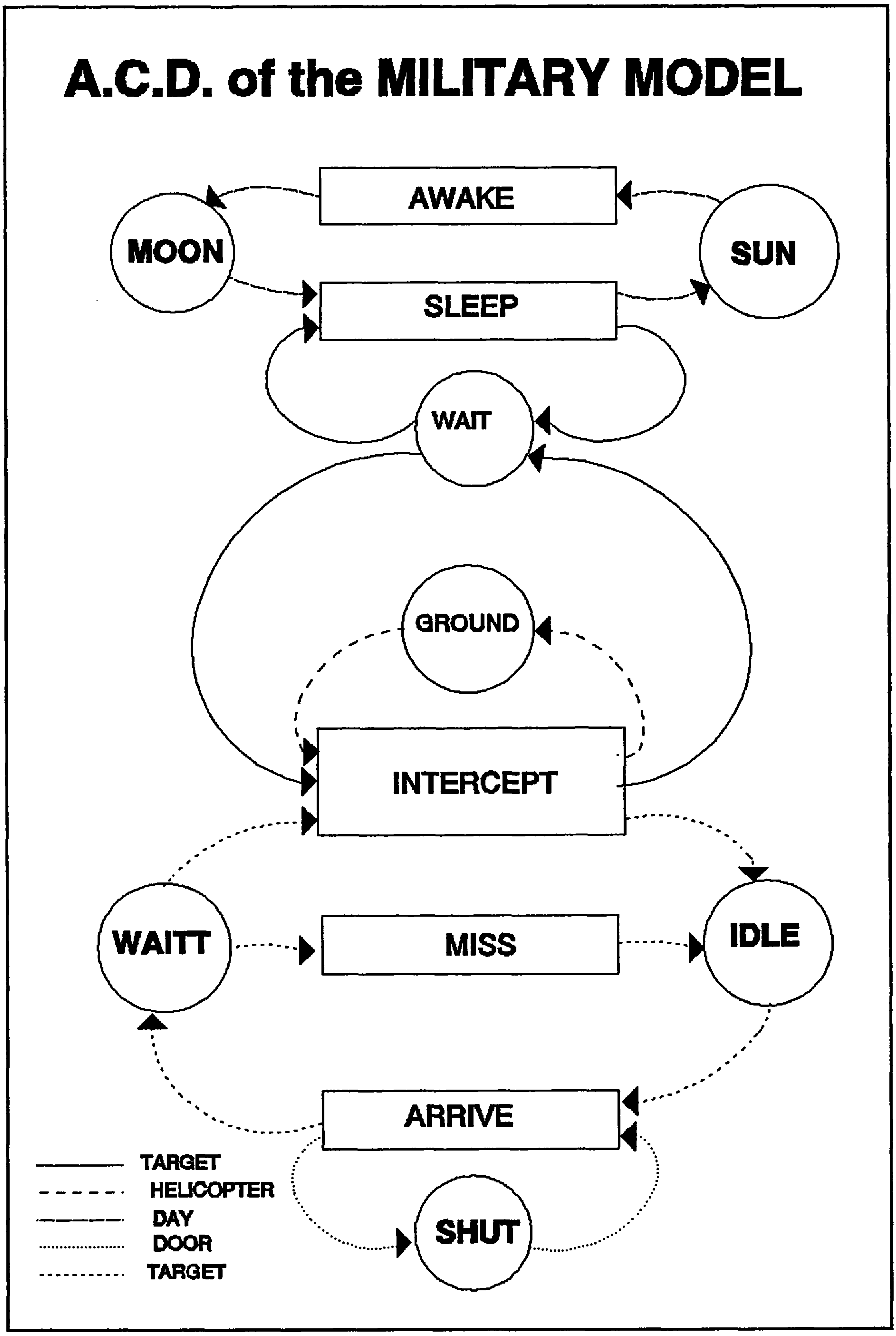



Figure A.10. A.C.D. of the MILITARY model.

### A.2.7. JACKSON'S MODEL

This problem has been taken from Winston (1987). For the sake of clarity it is described again in this section. Figure A.11. gives the corresponding A.C.D.

"The last two things that are done to a car before its manufacture is complete are installing the engine and putting on the tires. An average of 60 cars per hour arrive, and there is only one worker for the installation of the engine; he can serve an average of 54 cars per hour. There are three available workers for putting on the tyres, and each one can serve 162 cars per hour."

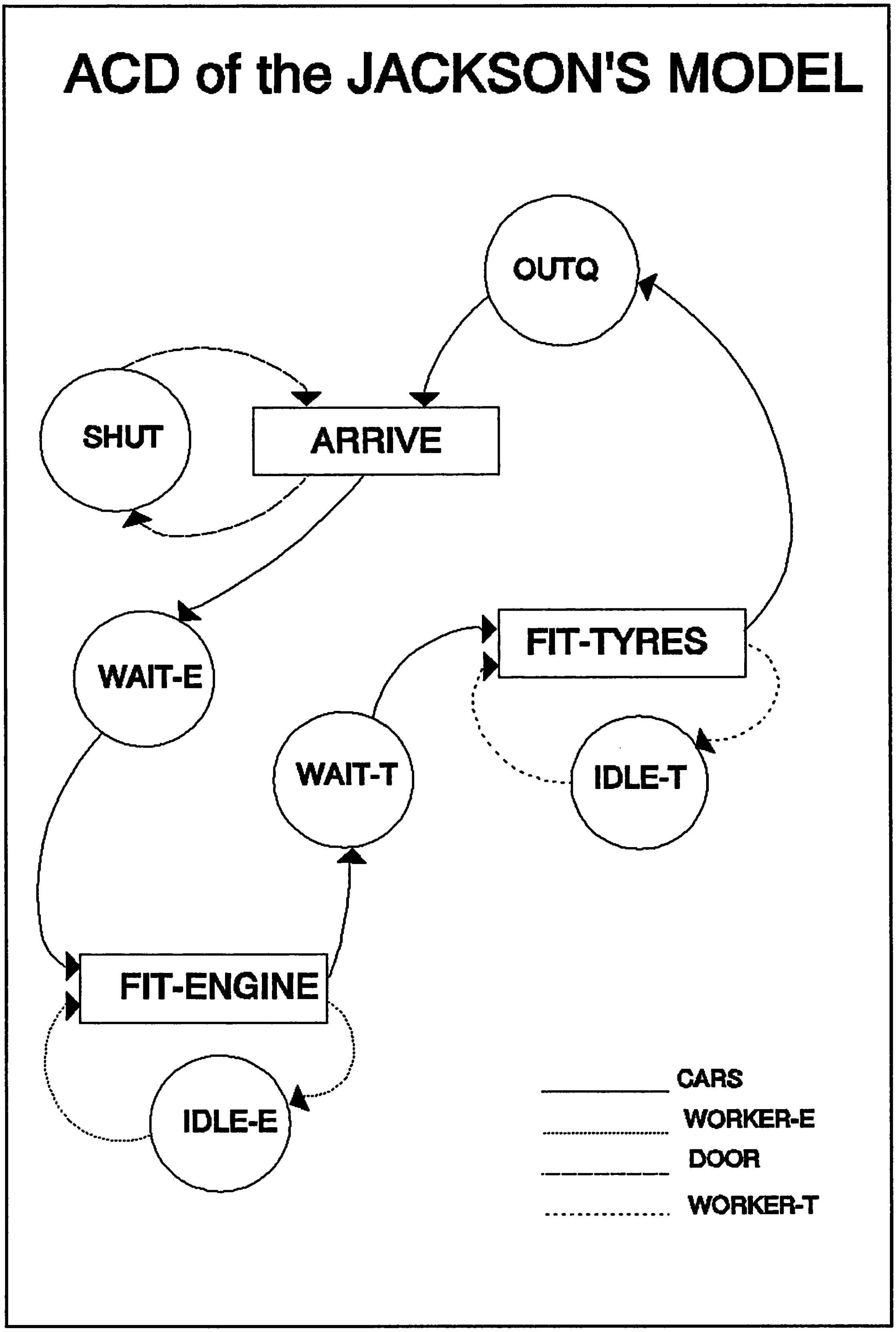



Figure A.11. A.C.D. of the Jackson's model.

# APPENDIX B : ADDITIONAL EXAMPLES FOR CHAPTER 2

## B.1. INTRODUCTION

Chapter 2 expands some of the ideas discussed very briefly in Chapter 1 and creates the scenario that is required for the reader to easily grasp the ideas and concepts of the following chapters.

Because we want to show in this thesis that a successful use of simulation does not require expensive to run and difficult to understand methods we discussed in Chapter 2 some problems which the simulation practitioner should deal with before the simulation is run. Among these problems we have that of identifying beforehand if the steady state for a particular system exists or not; if it does, it is important to carry out a further analysis previous to the simulation in order to identify possible "CRITICAL QUEUES" that may require a long simulation run length to reach the steady state. Some guidelines are proposed in Chapter 2 and in this appendix to help the practitioner in this identification. As was discussed in Chapter 2, the problem of critical queues is directly related to the values of the traffic intensity, $\tau = \lambda/(s\mu)$; when this value is greater than 1 there is no steady state, and the closer this value is to 1 the longer it takes for some of the queues in the system to reach the steady state. We will identify some possible reasons for this value to be greater than or very close to 1, and we will illustrate the theoretical reasoning with some simulation results.

## B.2. CRITICAL QUEUES: POSSIBLE REASONS FOR $\lambda/s\mu > 1$.

One of the reasons for the traffic intensity to be larger than 1 is the time taken by the activities to be executed:

*A relatively long execution time of one of the activities, or a short interarrival time, may cause instability of one or more of the queues belonging to temporary entities.*

The following example illustrates this point.

**THE FISH PACKING model.**

This is an example that shows the instability of some of the queues belonging to *temporary* entities due to a large execution time of one of the activities. Table B.1. gives four different sets of execution time for the activities ARRIVE, WEIGH and PACK. The system was simulated for each one of these cases; in some of them when the execution time of one of the activities is increased , or the interarrival time decreased, or both, one of the queues belonging to the **fish fillets** becomes unstable (only queues belonging to *temporary* entities can become unstable).

| ACTIVITY | CASE 1 | CASE 2 | CASE 3 | CASE 4 |
|---|---|---|---|---|
| ARRIVE | NEGEXP(10) | NEGEXP(5) | NEGEXP(10) | NEGEXP(10) |
| WEIGH | 2 | 2 | 4 | 2 |
| PACK | NORMAL(5,1) | NORMAL(5,1) | NORMAL(5,1) | NORMAL(10,2) |
| REPLACE | 1 | 1 | 1 | 1 |

**Table B.1.** Time taken to be executed by the different activities of the FISH PACKING model.

The WAIT and the ALLQ queues (See Figure A.8., Appendix A) belonging both to the FISH entity are studied in this example; for each one of these queues the mean queuing time was recorded for different simulation run lengths and for the different cases given in Table B.1; they are summarised in Tables B.2. and B.3. as follows:

| **Table** | **Queue** | **Parameter** |
|---|---|---|
| B.2. | WAIT | Queuing Time |
| B.3. | ALLQ | Queuing Time |

From Table B.2. (WAIT Queue) we notice that when the mean value of the **arrive** activity execution time is changed from 10 to 5 and, therefore, more **fish fillets** arrive to the system, the queuing time (and the queue length as well) reaches a steady state as $\tau = \lambda/s\mu = (1/5)/(1*(1/2)) = 2/5 < 1$ for the **weigh** activity. These two different experiments have been carried out for an execution time of the activity **weigh** of 2 units of time. However, if the mean value of the activity **arrive** is changed to 5 and at the same time the execution time of the activity **weigh** is increased to 6, the WAIT queue becomes unstable as $\tau = 6/5 > 1$ (mean estimates corresponding to this case are not given in Table B.2.)

| Run Length | Condition 1 | | Condition 2 | | Condition 3 | | Condition 4 | |
|---|---|---|---|---|---|---|---|---|
| | Mean | Std. Dev. | Mean | Std. Dev. | Mean | Std. Dev. | Mean | Std. |
| 1000 | 0.240 | 0.091 | 0.646 | 0.195 | 1.287 | 0.543 | 0.240 | 0.091 |
| 4000 | 0.244 | 0.047 | 0.670 | 0.083 | 1.295 | 0.259 | 0.244 | 0.047 |
| 7000 | 0.252 | 0.038 | 0.668 | 0.049 | 1.329 | 0.190 | 0.252 | 0.038 |
| 10000 | 0.249 | 0.028 | 0.671 | 0.047 | 1.317 | 0.138 | 0.249 | 0.028 |
| 13000 | 0.248 | 0.023 | 0.677 | 0.044 | 1.313 | 0.107 | 0.248 | 0.023 |
| 16000 | 0.250 | 0.022 | 0.672 | 0.038 | 1.329 | 0.108 | 0.250 | 0.022 |
| 19000 | 0.249 | 0.019 | 0.675 | 0.039 | 1.328 | 0.092 | 0.249 | 0.019 |
| 22000 | 0.251 | 0.018 | 0.676 | 0.033 | 1.331 | 0.088 | 0.251 | 0.018 |
| 25000 | 0.252 | 0.018 | 0.676 | 0.030 | 1.331 | 0.088 | 0.252 | 0.018 |
| 28000 | 0.252 | 0.018 | 0.678 | 0.028 | 1.332 | 0.081 | 0.252 | 0.018 |
| 29000 | 0.251 | 0.017 | 0.677 | 0.028 | 1.329 | 0.077 | 0.251 | 0.017 |
| 30000 | 0.251 | 0.016 | 0.677 | 0.026 | 1.329 | 0.075 | 0.251 | 0.016 |

**Table B.2.** Mean WAIT queuing time and standard deviation estimates as a function of the simulation run length for the four different conditions given in Table B.1.

From Table B.3. we notice that for conditions 2 and 4 the ALLQ queue will become unstable, and this is due to the fact that $\lambda/s\mu = 1$ for the PACK activity. In the results shown in Table B.3. the standard deviation of the mean estimates is also given. It can be seen that under conditions 2 and 4 the standard deviation will increase when the run length is increased. In general, from the empirical results recorded in this research, it was found that unless the standard deviation decreases when the run length is increased, the system does not reach the steady state. (See explanation for the LAUNDERETTE model, Chapter 2). As in the case of the LAUNDERETTE model, the instability of some of the queues is due to the large execution time of one of the activities as compared to that of the other activities.

| Run | Condition 1 | | Condition 2 | | Condition 3 | | Condition 4 | |
|---|---|---|---|---|---|---|---|---|
| | Mean | Std. | Mean | Std. | Mean | Std. | Mean | Std. |
| 1000 | 3.452 | 3.549 | 37.398 | 25.024 | 2.467 | 3.560 | 50.525 | 29.339 |
| 4000 | 3.130 | 1.386 | 85.624 | 37.927 | 2.143 | 1.361 | 113.638 | 68.755 |
| 7000 | 3.043 | 0.959 | 122.589 | 49.795 | 2.026 | 0.959 | 155.173 | 72.014 |
| 10000 | 3.049 | 0.764 | 151.024 | 67.110 | 2.037 | 0.775 | 198.433 | 79.446 |
| 13000 | 3.027 | 0.624 | 176.370 | 83.550 | 2.023 | 0.640 | 229.839 | 95.656 |
| 16000 | 3.006 | 0.537 | 201.090 | 98.901 | 1.993 | 0.548 | 258.363 | 108.771 |
| 19000 | 3.000 | 0.545 | 222.856 | 115.699 | 1.991 | 0.554 | 283.372 | 124.327 |
| 22000 | 2.971 | 0.481 | 242.320 | 129.773 | 1.960 | 0.490 | 309.261 | 139.015 |
| 25000 | 2.939 | 0.458 | 261.621 | 138.139 | 1.930 | 0.474 | 331.674 | 153.204 |
| 28000 | 2.949 | 0.444 | 282.208 | 148.181 | 1.940 | 0.457 | 355.291 | 166.705 |

**Table B.3.** Mean ALLQ queuing time and standard deviation estimates as a function of the simulation run length, and of the four different combinations of the activities execution time considered in Table B.1.

A second reason for the traffic intensity $\tau$ to take values greater than 1 is related to the number of units in the system of a permanent entity and can be expressed as follows:

*When the number of servers of a permanent entity is small, the value of the traffic intensity of those activities where this entity is involved gets larger, and one or more of the queues belonging to temporary entities may become unstable.*

The BRAZILIAN HOSPITAL and the PUB simulation models are used to illustrate this point.

**The BRAZILIAN HOSPITAL model.**

In this case (see Figure A.9., Appendix A) the WAIT queue may be unstable depending on the number of **receptionists** available in the RECQ queue. In this system, as defined by the problem, the activity **outpatient** has a priority over the activity **entry** and this one has a priority over the activity **reception**; as a consequence of this, the "customers" arriving to the reception will in general have to wait the longer. Due to this priority, when the number of receptionists in the Hospital is small (1 or 2 in this case) they will be mostly engaged with the **outpatient** and **entry** activities and therefore the patients arriving to the WAIT queue will take a long time to be serviced which may cause the queue length and the queuing time of this queue to increase without bound as the simulation run length increases. The above analysis can be done by simulation users with no knowledge of queuing theory. The theoretician with this knowledge can also identify the critical values of service rates, arrival rates and number of units of each entity that will make the system unstable. Figure B.1. compares the behaviour of the WAIT queue when there are 2 and 3 receptionists in RECQ. Two different Y-axis are used as the magnitude of the mean estimates for the two different number of receptionists greatly differ in value.

**The PUB model.**

In this case the WAIT queue for the entity **customer** (temporary entity) may show instability. The following example illustrates why and when the WAIT queue may become unstable.

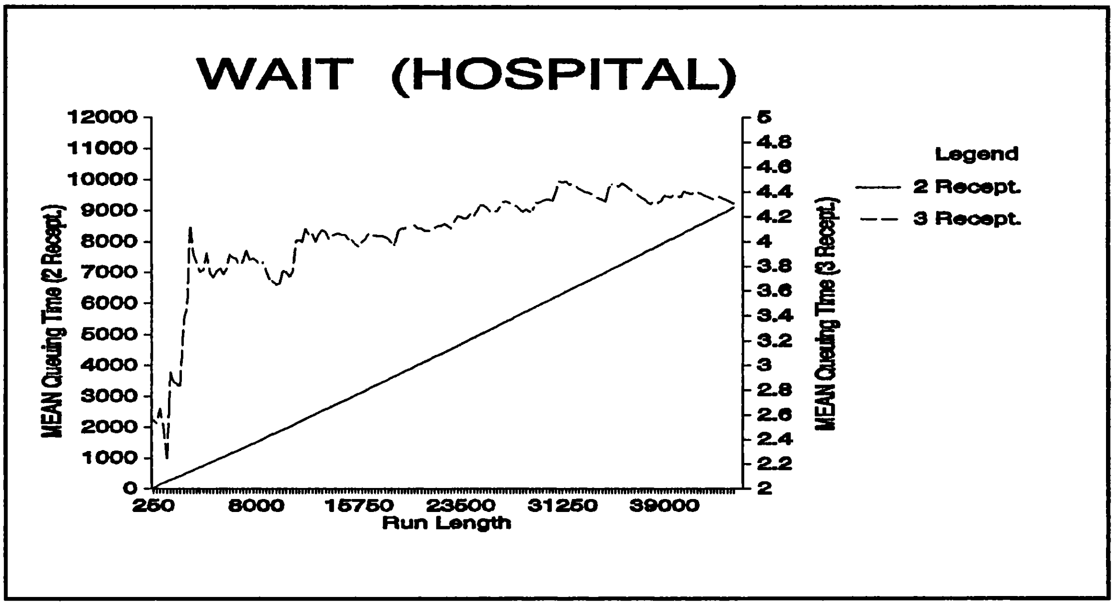



**Figure B.1.** Mean WAIT queuing time when there are 2 and 3 receptionists in the RECQ queue.

In the PUB model the **arrival** of customers is modelled using a negative exponential distribution with mean 15, and the POUR activity takes on average 6 minutes to be executed. In this example the factor that causes instability of the WAIT queue is the number of **barmaids** in the system. When this number is two or one the WAIT queue does not reach a steady state because $\lambda/s\mu$ = 15/12 or 15/6 which is greater than 1. A graph of the results for the WAIT queuing time is shown in Figure B.2. In this graph we compare the mean values for the WAIT queue as a function of the simulation run length when there is only one barmaid in the PUB with the same values when the number of barmaids is four.

The graph clearly show the instability of the WAIT queue in the case of one barmaid in the system.

## CONCLUSION

Queues that may not reach steady state are those belonging to temporary entities.

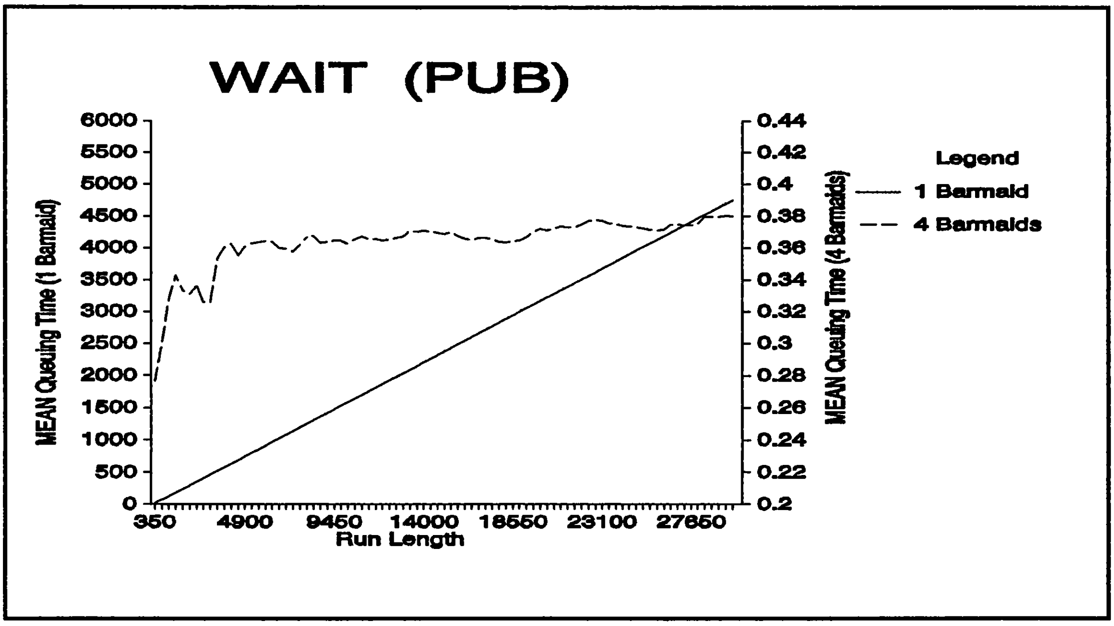



**Figure B.2.** Mean WAIT queuing time when there are 4 and 1 BARMAIDS in the IDLE queue.

This problem of not reaching the steady state, depends on the time that it takes to execute the activities where the temporary entity is involved. If one or more of these activities takes a long time to be executed as compared to the time of execution of the other activities, it may be possible that one of the queues of the temporary entity will never reach a steady state. Another possible cause of instability in one or more of the queues belonging to a temporary entity occurs when the temporary entity interacts with one or more permanent entities, with a small number of units in the system. For example, in the case of the PUB, the WAIT queue reaches or not a steady state depending on the number of barmaids in the system. This is due to the fact that the entity **customer** (to whom the WAIT queue belongs) requires of the entity **barmaid** to be poured a drink. When the number of barmaids is small they will not be able to cope at a given time with the incoming customers. The same is the case of the queue WAIT in the BRAZILIAN HOSPITAL system. In the case studied in this appendix a small number of receptionists will cause instability of this queue.

## B.3. QUEUES THAT MAY TAKE A RELATIVELY LONG TIME TO REACH THE STEADY STATE

In Chapter 2 we have identified some possible "critical" queues, that may require a long simulation run length to reach the steady state. We also showed in Chapter 3 how when we increase the number of replications used for the estimation of parameters of these critical queues, the steady state may be reached for shorter simulation run lengths. In this section we will give some additional examples of possible critical queues. We will also identify some other factors for which a queue may require a long simulation run length to reach the steady state. These are factors concerning the number of units of permanent entities, and also the execution time of some of the activities. In Section B.3.1. we will consider those possible critical queues that are due to the characteristics of the simulation models themselves, (i.e., we compare different queues under the same conditions) while in section B.3.2. we will study the influence of the number of units, the time the activities take to be executed, etc. and how a change in some of the parameters of the model can make a queue critical (i.e., we compare the same queue under different conditions).

### B.3.1. CRITICAL QUEUES DUE TO THE CHARACTERISTICS OF THE MODEL.

In this section we will give examples of queues that are critical due to some characteristics of the simulation model itself. Sometimes it is possible to identify them with an analysis previous to running the simulation. Some of the factors that we have identified in this study concern permanent entities, and others affect the behaviour of temporary entities. We will discuss very briefly each one of them, and we will illustrate them with an example of the LAUNDERETTE model.

## TEMPORARY ENTITIES

In most cases, the arrival of entities to a system are modelled using the negative exponential distribution. In this case, due to the skewness of this distribution, the queue in the simulation model to which the temporary entity "arrives" from the "outside world" will require a long simulation run length to reach the steady state. Identification of other critical queues belonging to temporary entities is not so straightforward as in this case we also need to take into account the permanent entities with which there is interaction.

## PERMANENT ENTITIES

A factor for which queues belonging to permanent entities may require a long simulation run length to reach the steady state is that of the number of units of the permanent entity. A small number of units, or of "servers" will increase the traffic intensity $\tau = \lambda/(s^*\mu)$ and sometimes as this value increases, the queue of the permanent entity will take longer to reach the steady state.

To illustrate these arguments, we use the LAUNDERETTE model. In this model, the arrival of customers to the system is modelled with a negative exponential distribution; therefore, we can expect the parameters of the WASHQ queue to require a long simulation run length to reach the steady state. Similarly, because there are 12 baskets in the system the BIDLE queue will possibly reach the steady state for a short simulation run length (the number of servers, s = 12, and this will give a small value of $\tau$; however, in analysing queues belonging to permanent entities that interact with several other entities we have to be cautious and no generalisation is possible). On the other hand, there are only 2 driers in the system, which will give a large value of $\tau$, and therefore the parameters of the DRIER queue will require a long simulation run length to reach the steady state.

Table B.4. gives the mean queuing time estimates as a function of the simulation run length for the following queues: WASHQ, BIDLE, DRYQ, DRIER and WMIDLE. In Appendix C we give close approximations to the real steady state value. As discussed in Chapter 2, we consider that the parameters have reached the steady state when the mean estimates all fall within 2.5% of this steady state value. These values, along with the ranges for which we can consider the parameters to be in the steady state are the following:

| Queue | Steady state value | Range |
|---|---|---|
| WASHQ | 6.675 | [6.508 , 6.842] |
| BIDLE | 67.260 | [65.578 , 68.94] |
| WMIDLE | 12.780 | [12.46 , 13.10] |
| DRYQ | 17.670 | [17.228 , 18.112] |
| DRIER | 1.993 | [1.943 , 2.04] |

In Table B.4. we have underlined those values for which we can consider the parameter to be in the steady state. From the underlined values we can see that as expected, the WASHQ, the DRYQ, and the DRIER queue require long simulation run lengths to reach the steady state (from the values in the table we notice that for a simulation run length as long as 22000 minutes the steady state has not been reached yet). The mean estimates of this table have been obtained from 100 replications. As discussed in Chapter 3, when we increase the number of replications, we can expect the steady state to be reached sooner. Similarly, from Table B.4. we notice that the BIDLE and the WMIDLE queues will reach the steady state for a short simulation run length compared to that required for the other queues that we have studied in this model. This was expected because of the number of units of these two entities as compared to the number of driers.

While in this section we compared different queues under the same conditions, in section B.3.2. we will compare the same queue under different conditions.

| Mean queuing time estimates | | | | | |
|---|---|---|---|---|---|
| Run Length | WASHQ | BIDLE | WMIDLE | DRYQ | DRIER |
| 6000 | 5.876 | 70.378 | 13.406 | 15.555 | 2.314 |
| 6500 | 6.005 | 69.938 | 13.272 | 15.794 | 2.274 |
| 7000 | 6.017 | 69.725 | 13.323 | 16.034 | 2.261 |
| 7500 | 6.031 | 69.461 | 13.221 | 16.139 | 2.223 |
| 8000 | 6.064 | 69.387 | 13.215 | 16.234 | 2.218 |
| 8500 | 5.999 | 69.348 | 13.242 | 16.251 | 2.205 |
| 9000 | 5.986 | 69.459 | 13.233 | 16.134 | 2.199 |
| 9500 | 6.061 | 69.417 | 13.211 | 16.161 | 2.195 |
| 10000 | 6.103 | 69.314 | 13.195 | 16.249 | 2.187 |
| 10500 | 6.108 | 69.304 | 13.181 | 16.225 | 2.177 |
| 11000 | 6.117 | 69.193 | 13.154 | 16.284 | 2.167 |
| 11500 | 6.193 | 68.983 | 13.078 | 16.397 | 2.153 |
| 12000 | 6.306 | 68.762 | 13.018 | 16.553 | 2.137 |
| 12500 | 6.353 | 68.594 | 13.009 | 16.697 | 2.126 |
| 13000 | 6.337 | 68.524 | 13.008 | 16.742 | 2.120 |
| 13500 | 6.382 | 68.446 | 13.002 | 16.807 | 2.114 |
| 14000 | 6.407 | 68.393 | 12.982 | 16.858 | 2.111 |
| 14500 | 6.406 | 68.416 | 13.007 | 16.855 | 2.112 |
| 15000 | 6.389 | 68.498 | 13.023 | 16.811 | 2.115 |
| 15500 | 6.364 | 68.543 | 13.009 | 16.748 | 2.111 |
| 16000 | 6.372 | 68.425 | 12.965 | 16.793 | 2.096 |
| 16500 | 6.366 | 68.320 | 12.952 | 16.859 | 2.089 |
| 17000 | 6.343 | 68.344 | 12.970 | 16.849 | 2.088 |
| 17500 | 6.307 | 68.433 | 13.008 | 16.806 | 2.094 |
| 18000 | 6.268 | 68.558 | 13.039 | 16.736 | 2.103 |
| 18500 | 6.234 | 68.654 | 13.057 | 16.679 | 2.108 |
| 19000 | 6.251 | 68.608 | 13.044 | 16.709 | 2.105 |
| 19500 | 6.291 | 68.506 | 13.013 | 16.760 | 2.097 |
| 20000 | 6.314 | 68.394 | 13.006 | 16.853 | 2.092 |
| 20500 | 6.323 | 68.363 | 12.996 | 16.880 | 2.089 |
| 21000 | 6.333 | 68.310 | 12.973 | 16.895 | 2.081 |
| 21500 | 6.315 | 68.304 | 12.986 | 16.909 | 2.082 |
| 22000 | 6.310 | 68.307 | 12.971 | 16.905 | 2.079 |

**Table B.4.** LAUNDERETTE mean queuing time estimates as a function of the simulation run length.

## B.3.2. CRITICAL QUEUES: INFLUENCE OF CHANGES IN THE CHARACTERISTICS OF THE MODEL

Sometimes a careful analysis of the A.C.D can give the practitioner an idea of which of the queues of the system will take a relatively long simulation run length to reach the steady state. In this and the following subsection we give some examples showing how a change in the conditions of a system may have an important influence on the simulation run length required for the parameters of some of the queues to reach the steady state. In this section we are interested in studying the change in the simulation run length required for a parameter to reach the steady state, for entities that are represented in the A.C.D. by a single queue. As was discussed in Section B.3.1, when the number of units of this type of entity is small, the queue may take a long time to reach the steady state. But while in Section B.3.1. we compared this queue to others of the same model, in the examples of this section (for the PUB and the FISH PACKING models) we will study the same queue under different conditions, i.e., for different number of units allocated at the beginning of the simulation to the single queue that is used to model the entity in the A.C.D.

### THE PUB model

This example was considered, but not explicitly discussed in Section 2.4.3. In this example, when the PUB was simulated for an arrival that follows a negative exponential distribution with mean 15, and the number of barmaids in the system is 3, the IDLE queue would not reach the steady state for simulation run lengths shorter than 20000 (See Table 2.3). In a second experiment, we changed the number of barmaids from 3 to 8. The arrival was modelled with a negative exponential distribution with mean 15. In this case, from the values of Table 2.5. the mean queuing time of the IDLE queue requires only a simulation run length of 2000 to reach the steady state. It is important to notice that the entity BARMAID is involved in two activities: WASH and POUR. However, in some cases when an entity that is represented by a single queue in the A.C.D. interacts only with another entity, and is involved in only one activity, like for example, the entity **scale** in the FISH PACKING model, the influence of a change in the number of units of the permanent entity (**scale** in this model) is not as noticeable as in the PUB case for the IDLE queue. The following example illustrates this point.

**THE FISH PACKING MODEL**

In this model, the entity **scale** is modelled by a single queue in the A.C.D (Figure A.8, Appendix A). If the entity is not engaged in the activity WEIGH it is idle in the queue.

The system was simulated for different number of scales in the system (1, 2 and 5), for an arrival rate of 7, and the results for the mean queuing time and the mean queue length estimates of the IDLE (scale entity) queue as a function of the simulation run length are given in Table B.5. From this table it can be seen that the steady state is reached for very short simulation run lengths independent of the number of scales; we know this by looking at the mean estimates and noticing that they all show convergence to a value that we may assume is very close to the real steady state value. For example, the mean queue length estimates are oscillating around the values 0.715 (1 scale), 4.714 (2 scales) and 4.715 (5 scales). In this case the effect of the number of scales on the simulation run length required for the queue to reach the steady state is not as noticeable as it is for the IDLE queue in the PUB example discussed above.

The main conclusion from this discussion is that although a small number of units of a permanent entity represented by a single queue in the A.C.D. may have an influence on the time required by the queue to reach the steady state, there are other factors and characteristics of the model that have to be considered. The basic principle of queuing theory that queues for which the traffic intensity ($\tau$) takes a value larger than 1 are unstable, and that the closer it gets to 1 the longer the simulation run length required for the queue to reach the steady state, is still valid, but only in the sense that one or more of the queues belonging to a temporary entity will be unstable; but as is shown in section 2.4.1., and to a lesser extent here, a careful analysis of the A.C.D. or of the characteristics of the system is necessary, to determine which queue(s) will appear never (or require a long time) to reach the steady state.

| IDLE (scale Entity) | | | | | | |
|---|---|---|---|---|---|---|
| Run Length | Mean Queue Length | | | Mean Queuing Time | | |
| | 1 Sc. | 2 Sc. | 5 Sc. | 1 Sc. | 2 Sc. | 5 Sc. |
| 50 | 0.679 | 1.653 | 4.585 | 5.156 | 10.856 | 20.997 |
| 100 | 0.714 | 1.712 | 4.712 | 6.019 | 13.200 | 29.058 |
| 150 | 0.716 | 1.714 | 4.713 | 5.458 | 12.400 | 31.188 |
| 200 | 0.713 | 1.713 | 4.713 | 5.225 | 12.135 | 31.306 |
| 250 | 0.714 | 1.714 | 4.713 | 5.172 | 12.072 | 31.486 |
| 300 | 0.716 | 1.716 | 4.716 | 5.190 | 12.184 | 32.160 |
| 350 | 0.720 | 1.719 | 4.719 | 5.274 | 12.353 | 32.627 |
| 400 | 0.719 | 1.718 | 4.718 | 5.185 | 12.243 | 32.695 |
| 450 | 0.713 | 1.712 | 4.712 | 5.027 | 11.948 | 32.143 |
| 500 | 0.712 | 1.712 | 4.712 | 5.024 | 11.921 | 32.108 |
| 1000 | 0.715 | 1.714 | 4.714 | 5.052 | 12.048 | 32.732 |
| 1500 | 0.714 | 1.714 | 4.714 | 5.014 | 11.987 | 32.710 |
| 2000 | 0.716 | 1.716 | 4.716 | 5.056 | 12.079 | 33.023 |
| 2500 | 0.716 | 1.716 | 4.716 | 5.050 | 12.075 | 33.023 |
| 3000 | 0.717 | 1.717 | 4.717 | 5.076 | 12.129 | 33.195 |
| 3500 | 0.716 | 1.716 | 4.716 | 5.039 | 12.062 | 33.062 |
| 4000 | 0.715 | 1.715 | 4.715 | 5.029 | 12.046 | 33.031 |
| 4500 | 0.715 | 1.715 | 4.715 | 5.021 | 12.029 | 32.988 |
| 5000 | 0.714 | 1.714 | 4.714 | 5.003 | 11.994 | 32.910 |
| 5500 | 0.713 | 1.713 | 4.713 | 4.987 | 11.966 | 32.852 |
| 6000 | 0.714 | 1.714 | 4.714 | 4.996 | 11.981 | 32.880 |
| 6500 | 0.714 | 1.714 | 4.714 | 4.995 | 11.980 | 32.894 |
| 7000 | 0.714 | 1.714 | 4.714 | 5.003 | 11.998 | 32.940 |
| 7500 | 0.715 | 1.715 | 4.715 | 5.015 | 12.022 | 33.006 |
| 8000 | 0.715 | 1.715 | 4.715 | 5.011 | 12.013 | 32.990 |

**Table B.5.** IDLE mean queuing time and mean queue length as a function of the simulation run length and of the number of scales (Sc.) in the system.

### B.3.3. ADDITIONAL CONSIDERATIONS

In this section we show how to proceed with the analysis of queues that in the A.C.D alternate with activities, forming a cycle activity - queue - activity... One example of such a situation is given by the analysis of the STEELWORKS where the queues belonging to the entity **torpedo** are studied as the number of torpedoes in the system is increased, keeping the number of units of the other entities constant.

#### THE STEELWORKS - NUMBER OF TORPEDOES VARIABLE

The system has been simulated for different number of torpedoes (3, 8 and 16); the number of **cranes** has been set to 2, the number of **pits** to 1 and the number of **steel furnaces** to 1.

Two queues of the entity **torpedo** are of interest, as they are the only ones to interact with other entities of the system: TBLOWQ and PITQ (Figure A.7., Appendix A). Because there is only one **pit** we can expect the queue length of the PITQ queue to increase as we increase the number of torpedoes, and as at the start of the simulation, due to the way the simulation has been programmed, the PITQ queue is **empty**, the change in the state of the queue will be large and the more the number of torpedoes in the system, the longer it will take for the parameters of the queue to reach the steady state.

A similar analysis can be done for the TBLOWQ queue. At this queue the torpedoes will wait for the **blast furnaces** (there are only two units of this entity in the system) to empty their blast (See Figure A.7, Appendix A). But due to the small number of blast furnaces, the more the number the torpedoes the more they will have to wait for the activity BLOW to be executed.

Figures B.3. and B.4. show the mean queuing time estimates of the TBLOWQ and of the PITQ respectively as a function of the simulation run length and of the number of torpedoes (3 and 16 torpedoes).

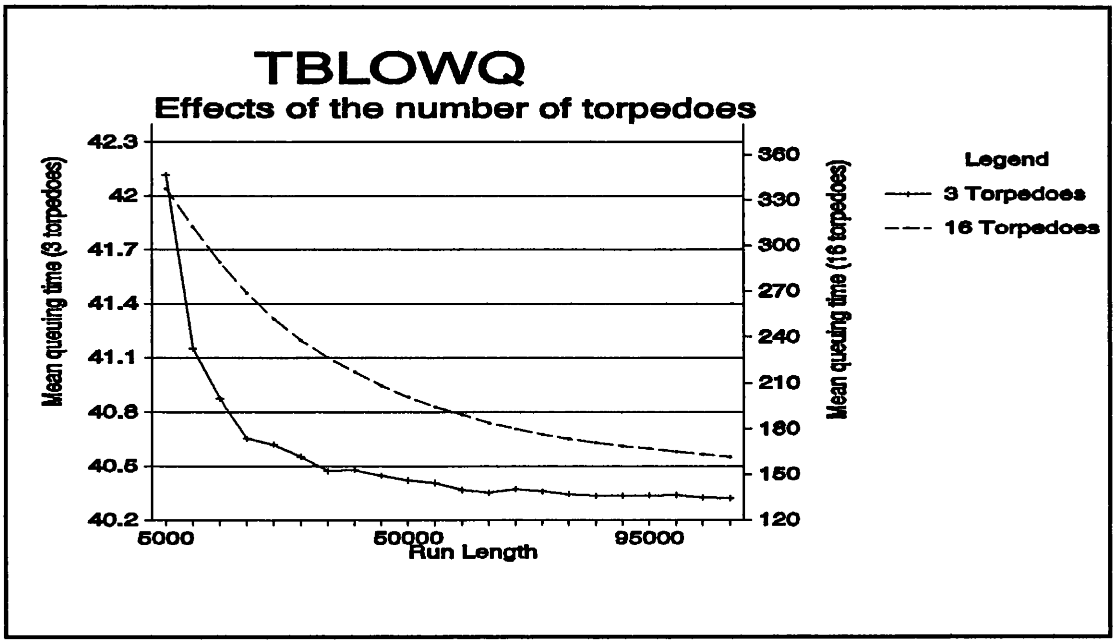



Figure B.3. TBLOWQ mean queuing time estimates as a function of the simulation run length and of the number of torpedoes.

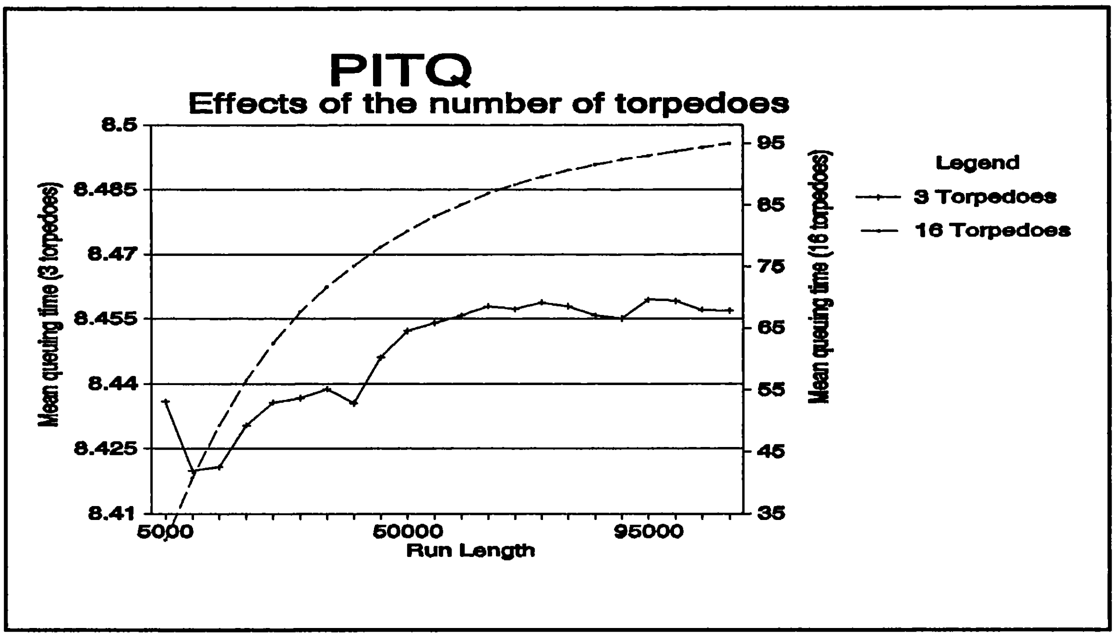



Figure B.4. PITQ mean queuing time estimates as a function of the simulation run length and of the number of torpedoes

From these figures we notice that when the number of torpedoes in the system is 3, there is clearly a convergence to the steady state value, but when there are 16 torpedoes in the system, the figures show still an increase (PITQ) or decrease (TBLOWQ) in the values of the mean estimates.

# APPENDIX C : STEADY STATE VALUES FOR SOME SIMULATION MODELS

## C.1. INTRODUCTION

Most of the research described in this thesis has been carried out in the area of steady state simulations; problems like, for example, that of the influence of the initial conditions and of the correlation among the observations when the batch means method is used are discussed and some new procedures are formulated. To test these new procedures, not only simple simulation models with known analytical answer but also more diverse simulation models are used. To make it easier the evaluation of the new procedures it is useful to obtain an accurate estimate of the real, but unknown, steady state value. This can be done by running the simulation model for an extremely long simulation run length and using a very large number of replications. This appendix contains the results thus obtained for the following simulation models:

1. The LAUNDERETTE.
2. The PUB.
3. The STEELWORKS model.
4. The MILITARY model.
5. The FISH PACKING model

The steady state mean value here estimated will be taken as the real mean value $\mu$; however, it is clear that in practical situations, due to the very long computer time that this approach requires, it would not feasible to do it and therefore, for practical purposes the practitioner expects that the estimates obtained from the simulation will be within a value $\epsilon$ of the real, but unknown value $\mu$. The value for this parameter $\epsilon$, that will be used to evaluate the accuracy of the results obtained using the new proposed procedures, is discussed in section C.2. Section C.3. presents the empirical results obtained for the different simulation models.

## C.2. DISCUSSION ON THE VALUE OF $\epsilon$ AS AN APPROXIMATION FOR THE REAL VALUE $\mu$ WHICH IS BEING ESTIMATED.

Usually when a new procedure is evaluated a simple model with known steady state value $\mu$ for the parameter(s) of interest is used, and the parameter is considered to be in the steady state when the simulation estimate falls within 0.5% (or even less) of $\mu$. In this research a larger value of $\epsilon = 2.5\%$ is used as a measure of the closeness of the estimate obtained from the simulation to the real value $\mu$. This choice of a larger value for this parameter is due to the fact that in those cases when simulation is used as a tool, estimating a queue length by 10 or by 10.2 (i.e. 2% larger) usually does not make any difference from a practical point of view. Obviously, in practice, this value of tolerance will depend of each project.

## C.3. STEADY STATE ESTIMATES FOR THE VALUE $\mu$.

We give in this section the mean estimates for the different queues of interest in this research. They were obtained using a very large number of replications and a very long simulation run length.

### C.3.1. THE LAUNDERETTE.

Tables C.1a and C.1b give the mean queuing time and the mean queue length steady state estimates as a function of the simulation run length, for the following queues:

1. WASHQ
2. BIDLE
3. WMIDLE
4. DRYQ
5. DRIER

| LAUNDERETTE - Mean queuing time estimates | | | | | |
|---|---|---|---|---|---|
| | WASHQ | BIDLE | WMIDLE | DRYQ | DRIER |
| Run Length | Mean | Mean | Mean | Mean | Mean |
| 700000 | 6.682 | 67.245 | 12.777 | 17.683 | 1.990 |
| 705000 | 6.681 | 67.246 | 12.776 | 17.680 | 1.990 |
| 710000 | 6.684 | 67.247 | 12.777 | 17.680 | 1.990 |
| 715000 | 6.683 | 67.254 | 12.778 | 17.674 | 1.990 |
| 720000 | 6.684 | 67.251 | 12.778 | 17.678 | 1.990 |
| 725000 | 6.685 | 67.249 | 12.778 | 17.680 | 1.990 |
| 730000 | 6.686 | 67.248 | 12.779 | 17.682 | 1.990 |
| 735000 | 6.684 | 67.258 | 12.780 | 17.675 | 1.991 |
| 740000 | 6.680 | 67.262 | 12.781 | 17.671 | 1.991 |
| 745000 | 6.677 | 67.268 | 12.782 | 17.667 | 1.991 |
| 750000 | 6.677 | 67.270 | 12.781 | 17.664 | 1.991 |
| 755000 | 6.677 | 67.264 | 12.780 | 17.666 | 1.991 |
| 760000 | 6.676 | 67.260 | 12.780 | 17.668 | 1.990 |
| 765000 | 6.677 | 67.261 | 12.780 | 17.671 | 1.990 |
| 770000 | 6.674 | 67.260 | 12.782 | 17.665 | 1.991 |
| 775000 | 6.672 | 67.263 | 12.782 | 17.666 | 1.991 |
| 780000 | 6.671 | 67.262 | 12.783 | 17.669 | 1.991 |
| 785000 | 6.671 | 67.260 | 12.782 | 17.671 | 1.991 |
| 790000 | 6.672 | 67.261 | 12.781 | 17.670 | 1.991 |
| 795000 | 6.671 | 67.262 | 12.781 | 17.672 | 1.991 |
| 800000 | 6.671 | 67.257 | 12.780 | 17.670 | 1.990 |

**Table C.1a.** Mean queuing time estimates for different queues of the LAUNDERETTE model given as a function of the simulation run length.

| LAUNDERETTE - Mean queuing time estimates | | | | | |
|---|---|---|---|---|---|
| | WASHQ | BIDLE | WMIDLE | DRYQ | DRIER |
| Run Length | Mean | Mean | Mean | Mean | Mean |
| 700000 | 0.836 | 8.412 | 1.598 | 2.212 | 0.249 |
| 705000 | 0.836 | 8.412 | 1.598 | 2.212 | 0.249 |
| 710000 | 0.836 | 8.412 | 1.598 | 2.212 | 0.249 |
| 715000 | 0.836 | 8.413 | 1.598 | 2.211 | 0.249 |
| 720000 | 0.836 | 8.413 | 1.598 | 2.212 | 0.249 |
| 725000 | 0.836 | 8.412 | 1.598 | 2.212 | 0.249 |
| 730000 | 0.836 | 8.412 | 1.599 | 2.212 | 0.249 |
| 735000 | 0.836 | 8.413 | 1.599 | 2.211 | 0.249 |
| 740000 | 0.836 | 8.414 | 1.599 | 2.211 | 0.249 |
| 745000 | 0.835 | 8.414 | 1.599 | 2.210 | 0.249 |
| 750000 | 0.835 | 8.415 | 1.599 | 2.210 | 0.249 |
| 755000 | 0.835 | 8.414 | 1.599 | 2.210 | 0.249 |
| 760000 | 0.835 | 8.414 | 1.599 | 2.210 | 0.249 |
| 765000 | 0.835 | 8.414 | 1.599 | 2.211 | 0.249 |
| 770000 | 0.835 | 8.414 | 1.599 | 2.210 | 0.249 |
| 775000 | 0.835 | 8.415 | 1.599 | 2.210 | 0.249 |
| 780000 | 0.835 | 8.415 | 1.599 | 2.210 | 0.249 |
| 785000 | 0.835 | 8.414 | 1.599 | 2.210 | 0.249 |
| 790000 | 0.835 | 8.413 | 1.599 | 2.211 | 0.249 |
| 795000 | 0.835 | 8.413 | 1.599 | 2.211 | 0.249 |
| 800000 | 0.835 | 8.413 | 1.599 | 2.211 | 0.249 |

**Table C.1b.** Mean queue length estimates for different queues of the LAUNDERETTE model given as a function of the simulation run length.

The data was obtained using 500 replications and a total simulation run length of 800000. From the above tables the following steady state values are determined:

| **Queue** | **Parameter** | **Steady state value ($\mu$)** |
|---|---|---|
| WASHQ | Queuing time | 6.675 |
| WASHQ | Queue Length | 0.835 |
| BIDLE | Queuing time | 67.260 |
| BIDLE | Queue length | 8.413 |
| WMIDLE | Queuing time | 12.780 |
| WMIDLE | Queue length | 1.599 |
| DRYQ | Queuing time | 17.670 |
| DRYQ | Queue length | 2.210 |
| DRIER | Queuing time | 1.990 |
| DRIER | Queue length | 0.249 |

### C.3.2. THE PUB

Using 500 replications and a total simulation run length of 800000, the mean queuing time and the mean queue length estimates for the WAIT, the CLEAN and the IDLE queues were obtained. Tables C.2a. (mean queuing time) and C.2b (mean queue length) show these estimates as a function of the simulation run length.

From these tables the following steady state values are determined:

| **Queue** | **Parameter** | **Steady state value ($\mu$)** |
|---|---|---|
| WAIT | Queuing time | 1.141 |
| WAIT | Queue length | 0.228 |
| CLEAN | Queuing time | 209.400 |
| CLEAN | Queue length | 41.881 |
| IDLE | Queuing time | 2.001 |
| IDLE | Queue length | 0.800 |

| PUB - Mean queuing time estimates | | | |
|---|---|---|---|
| Run Length | WAIT | CLEAN | IDLE |
| 700000 | 1.135 | 209.394 | 2.000 |
| 705000 | 1.138 | 209.380 | 2.000 |
| 710000 | 1.139 | 209.387 | 2.000 |
| 715000 | 1.140 | 209.395 | 2.001 |
| 720000 | 1.140 | 209.375 | 2.001 |
| 725000 | 1.140 | 209.380 | 2.001 |
| 730000 | 1.140 | 209.391 | 2.001 |
| 735000 | 1.140 | 209.371 | 2.000 |
| 740000 | 1.140 | 209.382 | 2.001 |
| 745000 | 1.141 | 209.402 | 2.001 |
| 750000 | 1.142 | 209.399 | 2.001 |
| 755000 | 1.142 | 209.386 | 2.001 |
| 760000 | 1.141 | 209.382 | 2.001 |
| 765000 | 1.141 | 209.370 | 2.001 |
| 770000 | 1.141 | 209.378 | 2.001 |
| 775000 | 1.141 | 209.395 | 2.001 |
| 780000 | 1.140 | 209.401 | 2.001 |
| 785000 | 1.140 | 209.412 | 2.001 |
| 790000 | 1.140 | 209.431 | 2.002 |
| 795000 | 1.141 | 209.406 | 2.001 |
| 800000 | 1.141 | 209.409 | 2.001 |

**Table C.2a.** Mean queuing time estimates for different queues of the PUB model given as a function of the simulation run length.

| PUB - Mean queue length estimates | | | |
|---|---|---|---|
| Run Length | WAIT | CLEAN | IDLE |
| 700000 | 0.228 | 41.884 | 0.800 |
| 705000 | 0.228 | 41.882 | 0.800 |
| 710000 | 0.228 | 41.882 | 0.800 |
| 715000 | 0.228 | 41.882 | 0.800 |
| 720000 | 0.228 | 41.879 | 0.800 |
| 725000 | 0.228 | 41.880 | 0.800 |
| 730000 | 0.228 | 41.881 | 0.800 |
| 735000 | 0.228 | 41.879 | 0.800 |
| 740000 | 0.228 | 41.879 | 0.800 |
| 745000 | 0.228 | 41.881 | 0.800 |
| 750000 | 0.228 | 41.880 | 0.800 |
| 755000 | 0.228 | 41.878 | 0.800 |
| 760000 | 0.228 | 41.878 | 0.800 |
| 765000 | 0.228 | 41.877 | 0.800 |
| 770000 | 0.228 | 41.878 | 0.800 |
| 775000 | 0.228 | 41.880 | 0.800 |
| 780000 | 0.228 | 41.881 | 0.800 |
| 785000 | 0.228 | 41.882 | 0.800 |
| 790000 | 0.228 | 41.883 | 0.800 |
| 795000 | 0.228 | 41.881 | 0.800 |
| 800000 | 0.228 | 41.882 | 0.800 |

**Table C.2b.** Mean queue length estimates for different queues of the PUB model given as a function of the simulation run length.

## C.3.3. THE STEELWORKS.

The system was simulated for a total simulation run length of 1300000 and replicated 350 times. Tables C.3a. and C.3b give the mean queuing time and the mean queue length estimates and their corresponding standard deviation as a function of the simulation run length.

| STEELWORKS - Mean queuing time estimates | | | | | | |
|---|---|---|---|---|---|---|
| Run Length | TBLOWQ | | PITQ | | LOADQ | |
| | Mean | Std.Dev. | Mean | Std.Dev. | Mean | Std.Dev. |
| 1200000 | 81.622 | 1.913 | 38.394 | 0.675 | 11.522 | 0.059 |
| 1205000 | 81.618 | 1.911 | 38.395 | 0.674 | 11.521 | 0.059 |
| 1210000 | 81.618 | 1.908 | 38.396 | 0.673 | 11.521 | 0.059 |
| 1215000 | 81.614 | 1.907 | 38.397 | 0.673 | 11.521 | 0.059 |
| 1220000 | 81.607 | 1.907 | 38.399 | 0.673 | 11.521 | 0.059 |
| 1225000 | 81.597 | 1.911 | 38.403 | 0.674 | 11.521 | 0.059 |
| 1230000 | 81.603 | 1.900 | 38.400 | 0.671 | 11.521 | 0.059 |
| 1235000 | 81.603 | 1.894 | 38.400 | 0.669 | 11.521 | 0.058 |
| 1240000 | 81.601 | 1.893 | 38.401 | 0.669 | 11.521 | 0.058 |
| 1245000 | 81.594 | 1.891 | 38.403 | 0.668 | 11.521 | 0.058 |
| 1250000 | 81.600 | 1.892 | 38.401 | 0.668 | 11.521 | 0.058 |
| 1255000 | 81.607 | 1.896 | 38.398 | 0.669 | 11.521 | 0.058 |
| 1260000 | 81.609 | 1.893 | 38.398 | 0.669 | 11.521 | 0.058 |
| 1265000 | 81.601 | 1.894 | 38.401 | 0.670 | 11.521 | 0.058 |
| 1270000 | 81.596 | 1.895 | 38.402 | 0.670 | 11.521 | 0.058 |
| 1275000 | 81.593 | 1.895 | 38.403 | 0.669 | 11.521 | 0.058 |
| 1280000 | 81.593 | 1.881 | 38.403 | 0.665 | 11.521 | 0.057 |
| 1285000 | 81.589 | 1.886 | 38.404 | 0.666 | 11.521 | 0.058 |
| 1290000 | 81.588 | 1.882 | 38.405 | 0.664 | 11.521 | 0.057 |
| 1295000 | 81.589 | 1.884 | 38.404 | 0.665 | 11.521 | 0.058 |
| 1300000 | 81.588 | 1.881 | 38.405 | 0.663 | 11.521 | 0.058 |

**Table C.3a.** Mean queuing time estimates for different queues of the STEELWORKS model given as a function of the simulation run length.

| STEELWORKS - Mean queue length estimates | | | | | | |
|---|---|---|---|---|---|---|
| Run Length | TBLOWQ | | PITQ | | LOADQ | |
| | Mean | Std.Dev. | Mean | Std.Dev. | Mean | Std.Dev. |
| 1200000 | 2.675 | 0.064 | 3.601 | 0.064 | 0.707 | 0.003 |
| 1205000 | 2.675 | 0.064 | 3.601 | 0.064 | 0.707 | 0.003 |
| 1210000 | 2.675 | 0.064 | 3.601 | 0.064 | 0.707 | 0.003 |
| 1215000 | 2.675 | 0.064 | 3.601 | 0.064 | 0.707 | 0.003 |
| 1220000 | 2.675 | 0.064 | 3.601 | 0.064 | 0.707 | 0.003 |
| 1225000 | 2.674 | 0.064 | 3.602 | 0.064 | 0.707 | 0.003 |
| 1230000 | 2.674 | 0.064 | 3.601 | 0.064 | 0.707 | 0.003 |
| 1235000 | 2.674 | 0.063 | 3.601 | 0.063 | 0.707 | 0.003 |
| 1240000 | 2.674 | 0.063 | 3.601 | 0.063 | 0.707 | 0.003 |
| 1245000 | 2.674 | 0.063 | 3.602 | 0.063 | 0.707 | 0.003 |
| 1250000 | 2.674 | 0.063 | 3.601 | 0.063 | 0.707 | 0.003 |
| 1255000 | 2.675 | 0.063 | 3.601 | 0.063 | 0.707 | 0.003 |
| 1260000 | 2.675 | 0.063 | 3.601 | 0.063 | 0.707 | 0.003 |
| 1265000 | 2.674 | 0.063 | 3.601 | 0.063 | 0.707 | 0.003 |
| 1270000 | 2.674 | 0.063 | 3.602 | 0.063 | 0.707 | 0.003 |
| 1275000 | 2.674 | 0.063 | 3.602 | 0.063 | 0.707 | 0.003 |
| 1280000 | 2.674 | 0.063 | 3.602 | 0.063 | 0.707 | 0.003 |
| 1285000 | 2.674 | 0.063 | 3.602 | 0.063 | 0.707 | 0.003 |
| 1290000 | 2.674 | 0.063 | 3.602 | 0.063 | 0.707 | 0.003 |
| 1295000 | 2.674 | 0.063 | 3.602 | 0.063 | 0.707 | 0.003 |
| 1300000 | 2.674 | 0.063 | 3.602 | 0.063 | 0.707 | 0.003 |

**Table C.3b.** Mean queue length estimates for different queues of the STEELWORKS model given as a function of the simulation run length.

From these tables we obtain the following values for the steady state mean queuing time and mean queue length for the different queues of interest in this model:

| Queue | Parameter | Steady state value ($\mu$) |
|---|---|---|
| TBLOWQ | Queuing time | 81.558 |
| TBLOWQ | Queue length | 2.674 |
| PITQ | Queuing time | 38.404 |
| PITQ | Queue length | 3.602 |
| LOADQ | Queuing time | 11.521 |
| LOADQ | Queue length | 0.707 |

## C.3.4. THE MILITARY MODEL

This simulation model was simulated for a total simulation run length of 1000000 using 300 replications. Two queues: WAIT and GROUND are studied. Tables C.4a. and C.4b give the mean queuing time and mean queue length estimates and their corresponding standard deviation as a function of the simulation run length.

| MILITARY MODEL - Mean queuing time estimates | | | | |
|---|---|---|---|---|
| | WAIT | | GROUND | |
| Run Length | Mean | Std. Dev. | Mean | Std. Dev. |
| 200000 | 2.872 | 0.040 | 19.378 | 0.068 |
| 400000 | 2.835 | 0.031 | 19.321 | 0.053 |
| 600000 | 2.803 | 0.026 | 19.270 | 0.037 |
| 800000 | 2.781 | 0.021 | 19.253 | 0.031 |
| 1000000 | 2.768 | 0.018 | 19.246 | 0.026 |
| 1200000 | 2.759 | 0.015 | 19.239 | 0.024 |
| 1400000 | 2.753 | 0.014 | 19.235 | 0.022 |
| 1600000 | 2.748 | 0.013 | 19.233 | 0.020 |
| 1800000 | 2.744 | 0.012 | 19.230 | 0.019 |
| 1900000 | 2.742 | 0.011 | 19.229 | 0.018 |
| 2000000 | 2.741 | 0.011 | 19.228 | 0.017 |

**Table C.4a.** Mean queuing time estimates for different queues of the MILITARY model given as a function of the simulation run length.

| MILITARY MODEL - Mean queue length estimates | | | | |
|---|---|---|---|---|
| | WAIT | | GROUND | |
| Run Length | Mean | Std. Dev. | Mean | Std. Dev. |
| 200000 | 0.180 | 0.002 | 1.178 | 0.002 |
| 400000 | 0.178 | 0.002 | 1.175 | 0.002 |
| 600000 | 0.177 | 0.002 | 1.173 | 0.001 |
| 800000 | 0.175 | 0.001 | 1.173 | 0.001 |
| 1000000 | 0.174 | 0.001 | 1.172 | 0.001 |
| 1200000 | 0.174 | 0.001 | 1.172 | 0.001 |
| 1400000 | 0.174 | 0.001 | 1.172 | 0.001 |
| 1600000 | 0.173 | 0.001 | 1.172 | 0.001 |
| 1800000 | 0.173 | 0.001 | 1.172 | 0.001 |
| 1900000 | 0.173 | 0.001 | 1.172 | 0.001 |
| 2000000 | 0.173 | 0.001 | 1.172 | 0.001 |

**Table C.4b.** Mean queue length estimates for different queues of the MILITARY model given as a function of the simulation run length.

To obtain these values, 5 days of real time continuous computer operation were spent. As can be seen, the mean queuing time estimates do not show a clear convergence to a value. However, as the real value was not used in the examples of this model to determine the minimum simulation run length for which the mean estimates fall within 2.5% of this real value, we do not increased the run length to obtain a more accurate estimate.

From these tables the following approximate steady state values are determined:

| **Queue** | **Parameter** | **Steady state value ($\mu$)** |
|---|---|---|
| WAIT | Queuing time | 2.740 |
| WAIT | Queue length | 0.173 |
| GROUND | Queuing time | 19.220 |
| GROUND | Queue length | 1.172 |

## C.3.5. THE FISH PACKING MODEL

The steady state mean queuing time and mean queue length estimates for the WAIT, ALLQ and IDLE queues (see ACD, Appendix A) have been estimated for this model. The time taken by the different activities to be executed is as follows:

| Activity | Execution time (distribution) |
|---|---|
| ARRIVE | NEGEXP(6) |
| WEIGH | 2 |
| PACK | NORMAL(5,1) |
| REPLACE | 2 |

In order to obtain accurate estimates of the mean queuing time and the mean queue length the model was simulated for a total simulated time of 800000 minutes using 300 replications. Table C.5a. gives the mean queuing time estimates and Table C.5b the mean queue length estimates for the three different queues of interest in this model.

From these tables the following steady state mean values can be estimated:

| Queue | Parameter | Steady state value ($\mu$) |
|---|---|---|
| WAIT | Queuing time | 0.511 |
| WAIT | Queue length | 0.085 |
| ALLQ | Queuing time | 14.900 |
| ALLQ | Queue length | 2.526 |
| IDLE | Queuing time | 3.990 |
| IDLE | Queue length | 0.666 |

| FISH PACKING - Mean queuing time estimates | | | |
|---|---|---|---|
| Run Length | WAIT | ALLQ | IDLE |
| 150000 | 0.510 | 14.736 | 3.997 |
| 200000 | 0.511 | 14.820 | 3.996 |
| 250000 | 0.511 | 14.814 | 3.995 |
| 300000 | 0.511 | 14.804 | 3.996 |
| 350000 | 0.511 | 14.837 | 3.995 |
| 400000 | 0.511 | 14.840 | 3.994 |
| 450000 | 0.511 | 14.873 | 3.993 |
| 500000 | 0.511 | 14.869 | 3.993 |
| 550000 | 0.511 | 14.891 | 3.993 |
| 600000 | 0.511 | 14.898 | 3.993 |
| 650000 | 0.511 | 14.901 | 3.993 |
| 700000 | 0.511 | 14.885 | 3.993 |
| 750000 | 0.511 | 14.899 | 3.993 |
| 800000 | 0.511 | 14.909 | 3.993 |

**Table C.5a.** WAIT, ALLQ and IDLE mean queuing time estimates as a function of the simulation run length.

| FISH PACKING - Mean queue length estimates | | | |
|---|---|---|---|
| Run Length | WAIT | ALLQ | IDLE |
| 150000 | 0.085 | 2.498 | 0.667 |
| 200000 | 0.085 | 2.513 | 0.666 |
| 250000 | 0.085 | 2.512 | 0.666 |
| 300000 | 0.085 | 2.510 | 0.666 |
| 350000 | 0.085 | 2.516 | 0.666 |
| 400000 | 0.085 | 2.517 | 0.666 |
| 450000 | 0.085 | 2.523 | 0.666 |
| 500000 | 0.085 | 2.522 | 0.666 |
| 550000 | 0.085 | 2.526 | 0.666 |
| 600000 | 0.085 | 2.527 | 0.666 |
| 650000 | 0.085 | 2.528 | 0.666 |
| 700000 | 0.085 | 2.525 | 0.666 |
| 750000 | 0.085 | 2.527 | 0.666 |
| 800000 | 0.085 | 2.529 | 0.666 |

**Table C.5b.** WAIT, ALLQ and IDLE mean queue length estimates as a function of the simulation run length.

# APPENDIX D : ADDITIONAL RESULTS CORRESPONDING TO THE SIMULATION MODELS USED IN CHAPTER 3.

## D.1. INTRODUCTION

In Chapter 3 we showed how, based on the *double sequential* approach described in the literature, it is possible to estimate the number of replications to be used for the estimation of a particular parameter of a simulation model. This estimation can be done using data obtained for a short simulation run length, and the number of replications thus estimated when used for the estimation of steady state parameters will give a curve of the mean estimates as a function of the simulation run length which is a good approximation to the real, but unknown, one. In this way it is easier to estimate the simulated time for which the curve becomes horizontal. In this appendix we include some additional empirical results obtained for other simulation models that confirm the above points.

## D.2. ANALYSIS OF THE EMPIRICAL RESULTS OBTAINED FOR SOME SIMULATION MODELS.

In this subsection we discuss the results obtained for the mean queuing time of some queues of the following simulation models: the LAUNDERETTE, the FISH PACKING and the STEELWORKS.

The study is divided into two parts: the estimation of the number of replications and the evaluation of the performance of this estimated number. A good performance means that the point in simulated time for which the curve of the mean estimates becomes horizontal is easier to determine if we use the number of replications estimated following the procedure discussed in Chapter 3 than if less replications are used; however, an increase in the number of replication will not make an appreciable difference in the shape of the curve.

## D.2.1. THE LAUNDERETTE MODEL : ESTIMATION OF THE NUMBER OF REPLICATIONS.

Three queues of this model have been studied (see Appendix A, Figure A.5.): the WASHQ, the BIDLE and the DRYQ queues. Using the double sequential approach described in Section 3.2. we obtained the following estimates for the number of replications to be used for the estimation of the steady state mean queuing time of the queues of interest:

| QUEUE | No. of Replications |
|---|---|
| WASHQ | 900 |
| BIDLE | 200 |
| DRYQ | 400 |

## D.2.2. THE LAUNDERETTE MODEL : EVALUATION OF THE PERFORMANCE OF THE PROPOSED PROCEDURE

To show how the use of the number of replications estimated in section D.2.1. for each one of the different queues of interest in this model gives good results we obtained the mean estimates as a function of the simulation run length and of the number of replications. Figures D.1., D.2. and D.3. show the mean estimates as a function of the simulation run length for the WASHQ, the BIDLE and the DRYQ queues.

From these figures it is easy to estimate the following simulation run lengths for which the curve becomes horizontal:

| Queue | Simulation run length |
|---|---|
| WASHQ | 9500 |
| BIDLE | 7500 |
| DRYQ | 11000 |

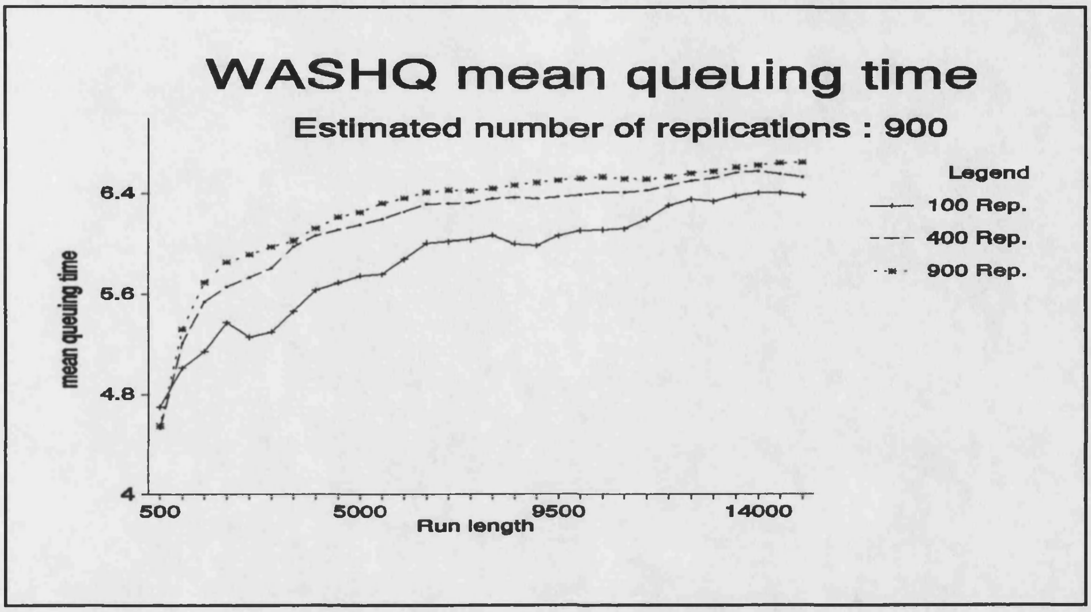



**Figure D.1.** WASHQ mean queuing time estimates as a function of the simulation run length and of the number of replications.

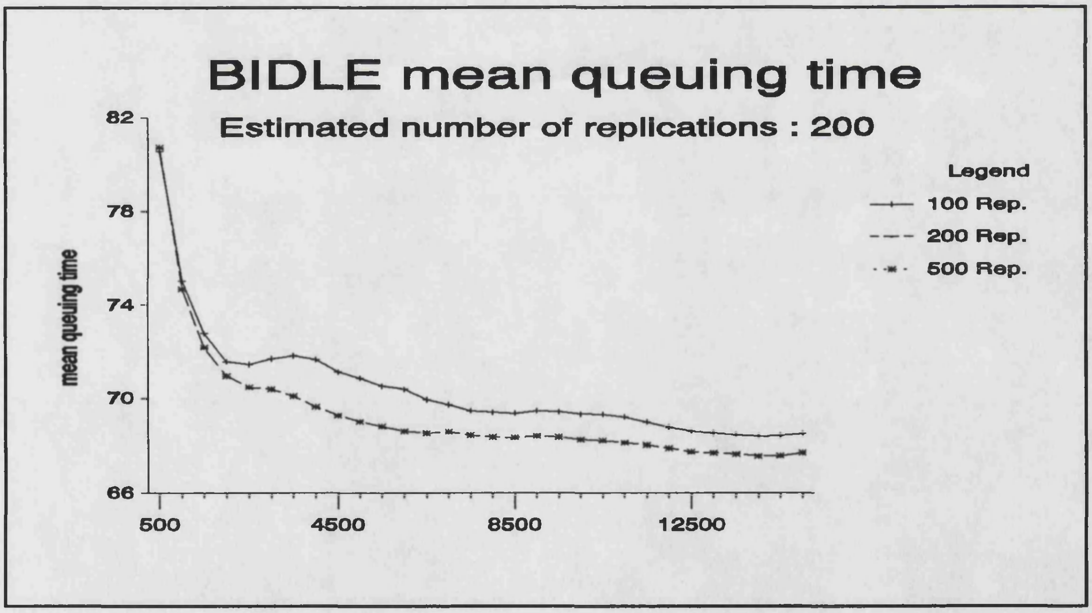



**Figure D.2.** BIDLE mean queuing time estimates as a function of the simulation run length and of the number of replications.

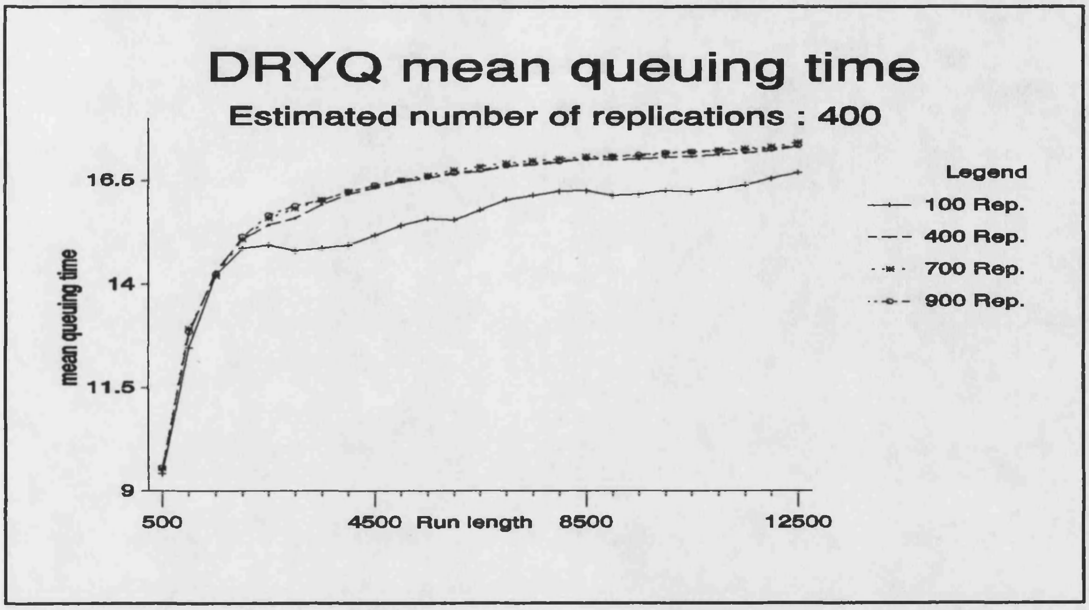



**Figure D.3.** DRYQ mean queuing time estimates as a function of the simulation run length and of the number of replications.

To show the influence of the number of replications on the c.i. width we give in tables D.1.a, D.1.b, and D.1.c. the c.i. half-width and the c.i. lower and upper limits as a function of the number of replications for different simulation run lengths and for the mean queuing time parameter of each one of the queues of interest.

From the tables we note how the c.i. obtained for the different queues, for simulation run lengths longer than those for which the curve of the mean estimates becomes horizontal, cover the real steady state value, which was determined in an empirical way in Appendix C. These steady state values are repeated here for the sake of convenience:

| QUEUE | Steady state ($\mu$) |
|---|---|
| WASHQ | 6.675 |
| BIDLE | 67.260 |
| DRYQ | 17.670 |

| WASHQ mean queuing time | | | | | |
|---|---|---|---|---|---|
| Results based on 150 replications | | | | | |
| Run Length | Mean | Std.Dev. | c.i. half-width | c.i.l.l. | c.i.u.l. |
| 7000 | 6.26 | 3.03 | 0.49 | 5.78 | 6.75 |
| 13000 | 6.48 | 2.51 | 0.40 | 6.08 | 6.89 |
| 19000 | 6.35 | 2.01 | 0.32 | 6.03 | 6.67 |
| Results based on 500 replications | | | | | |
| Run Length | Mean | Std.Dev. | c.i. half-width | c.i.l.l. | c.i.u.l. |
| 7000 | 6.32 | 2.97 | 0.26 | 6.06 | 6.58 |
| 13000 | 6.53 | 2.42 | 0.21 | 6.31 | 6.74 |
| 19000 | 6.53 | 2.13 | 0.19 | 6.35 | 6.72 |
| Results based on 900 replications | | | | | |
| Run Length | Mean | Std.Dev. | c.i. half-width | c.i.l.l. | c.i.u.l. |
| 7000 | 6.42 | 3.37 | 0.22 | 6.20 | 6.64 |
| 13000 | 6.58 | 2.61 | 0.17 | 6.41 | 6.75 |
| 19000 | 6.63 | 2.27 | 0.15 | 6.49 | 6.78 |

**Table D.1.a.** WASHQ c.i. width, c.i. lower and upper limits for different number of replications and different simulation run lengths.

| BIDLE mean queuing time parameter | | | | | |
|---|---|---|---|---|---|
| Results based on 200 replications | | | | | |
| Run Length | Mean | Std.Dev. | c.i. half-width | c.i.l.l. | c.i.u.l. |
| 7000 | 69.10 | 8.36 | 1.34 | 67.76 | 70.43 |
| 13000 | 68.16 | 6.49 | 1.04 | 67.12 | 69.20 |
| 19000 | 68.25 | 5.54 | 0.89 | 67.37 | 69.14 |
| Results based on 400 replications | | | | | |
| Run Length | Mean | Std.Dev. | c.i. half-width | c.i.l.l. | c.i.u.l. |
| 7000 | 68.44 | 8.29 | 0.73 | 67.72 | 69.17 |
| 13000 | 67.58 | 6.35 | 0.56 | 67.03 | 68.14 |
| 19000 | 67.60 | 5.49 | 0.48 | 67.12 | 68.08 |
| Results based on 900 replications | | | | | |
| Run Length | Mean | Std.Dev. | c.i. half-width | c.i.l.l. | c.i.u.l. |
| 7000 | 68.55 | 8.87 | 0.58 | 67.97 | 69.13 |
| 13000 | 67.71 | 6.61 | 0.43 | 67.28 | 68.15 |
| 19000 | 67.55 | 5.61 | 0.37 | 67.18 | 67.91 |

**Table D.1.b.** BIDLE c.i. width, and lower and upper c.i. limits for different number of replications and different simulation run lengths.

| DRYQ mean queuing time parameter | | | | | |
|---|---|---|---|---|---|
| Results based on 150 replications | | | | | |
| Run Length | Mean | Std.Dev. | c.i. half-width | c.i.l.l. | c.i.u.l. |
| 7000 | 16.49 | 6.04 | 0.97 | 15.52 | 17.45 |
| 13000 | 17.05 | 4.69 | 0.75 | 16.30 | 17.80 |
| 19000 | 16.98 | 3.93 | 0.63 | 16.35 | 17.61 |
| Results based on 400 replications | | | | | |
| Run Length | Mean | Std.Dev. | c.i. half-width | c.i.l.l. | c.i.u.l. |
| 7000 | 16.80 | 5.93 | 0.52 | 16.28 | 17.32 |
| 13000 | 17.38 | 4.58 | 0.40 | 16.98 | 17.78 |
| 19000 | 17.41 | 3.95 | 0.35 | 17.06 | 17.75 |
| Results based on 700 replications | | | | | |
| Run Length | Mean | Std.Dev. | c.i. half-width | c.i.l.l. | c.i.u.l. |
| 7000 | 16.85 | 6.35 | 0.41 | 16.44 | 17.27 |
| 13000 | 17.39 | 4.74 | 0.31 | 17.08 | 17.70 |
| 19000 | 17.49 | 4.02 | 0.26 | 17.23 | 17.75 |

**Table D.1.c.** DRYQ c.i. width and lower and upper c.i. limits for different simulation run lengths and different number of replications.

### D.2.3. THE FISH PACKING MODEL : ESTIMATION OF THE NUMBER OF REPLICATIONS

Three queues of this model have been studied: the WAIT, the IDLE, and the ALLQ queues (see Figure A.8., Appendix A). Following the sequential approach we obtained the following estimates for the number of replications to be used for the estimation of the mean queuing time of each one of the queues:

| QUEUE | No. of Replications |
|---|---|
| WAIT | 75 |
| ALLQ | 600 |
| IDLE | 75 |

### D.2.4. THE FISH PACKING MODEL : EVALUATION OF THE PERFORMANCE OF THE PROPOSED PROCEDURE

Figures D.4, D.5 and D.6. give the mean queuing time estimates for the WAIT, the IDLE and the ALLQ queues respectively, as a function of the simulation run length and of the number of replications.

In Appendix C we showed that the steady state mean queuing time values for these queues are:

| Queue | Steady state ($\mu$) |
|---|---|
| WAIT | 0.511 |
| IDLE | 3.990 |
| ALLQ | 14.900 |

Although from a statistical point of view, as discussed in Section 2.6.1., we should not use the WAIT mean as an estimator due to the small steady state value, we show the mean queuing time estimates in Figure D.4. to show how, even in cases like this, the procedure we propose gives good results.

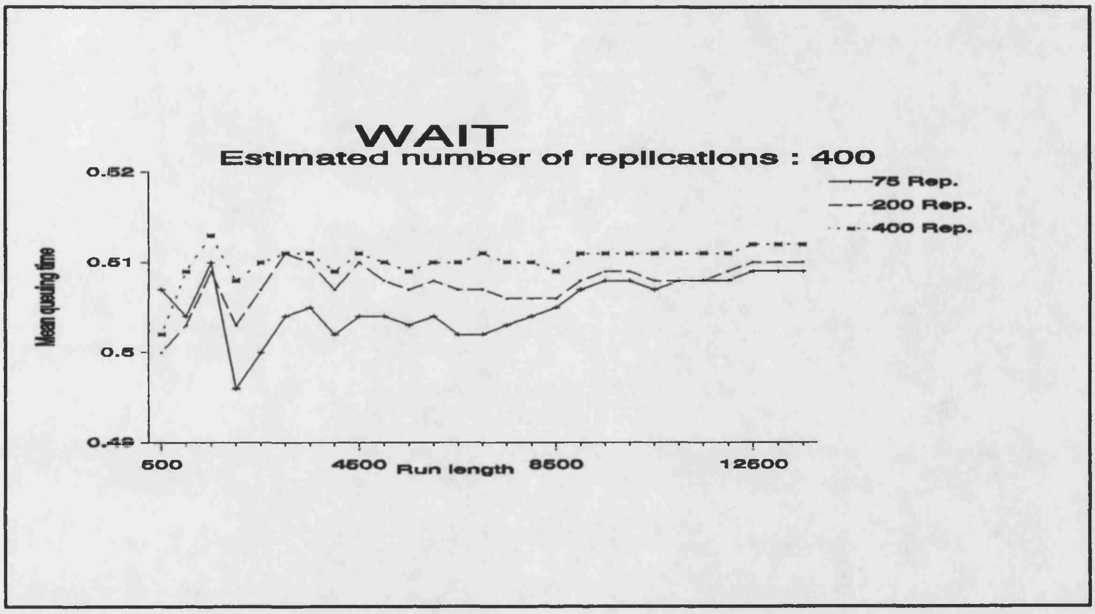



**Figure D.4.** WAIT mean queuing time estimates as a function of the simulation run length and of the number of replications.

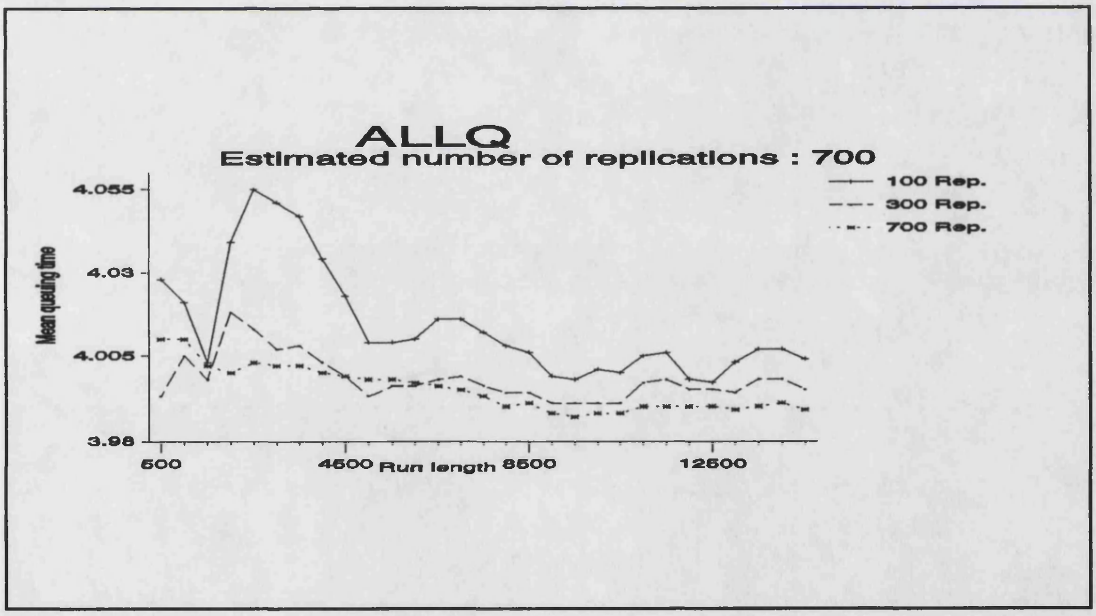



**Figure D.5.** ALLQ mean queuing time estimates as a function of the simulation run length and of the number of replications.

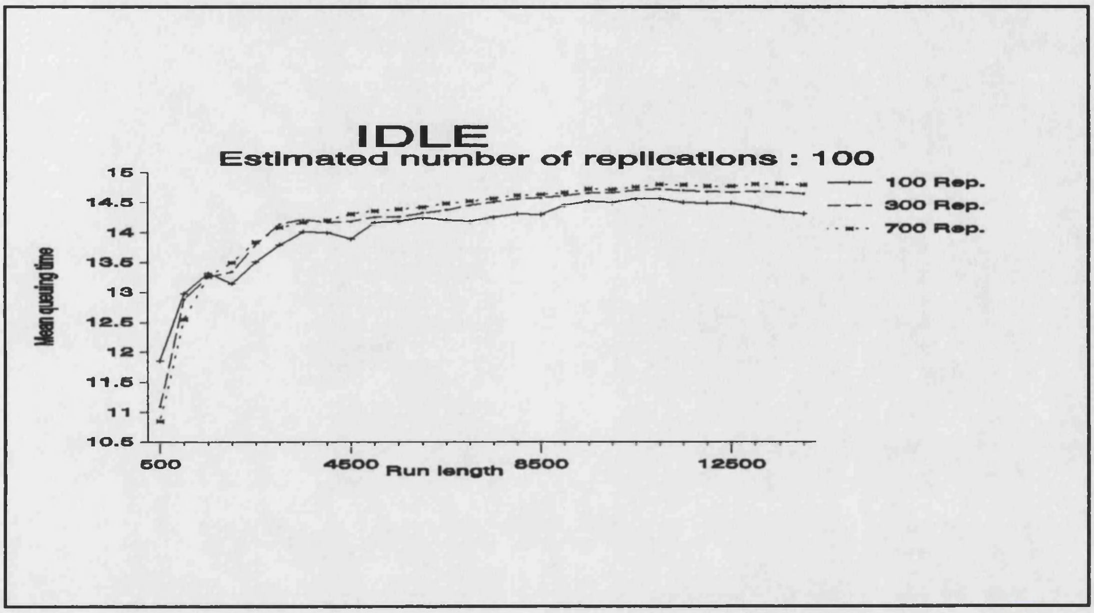



**Figure D.6.** IDLE mean queuing time estimates as a function of the simulation run length and of the number of replications.

From these figures we note how when the number of replications estimated in section D.2.3. is used it is easy to estimate the simulated time for which the curve becomes horizontal. This time is the following for the queues of interest:

| **QUEUE** | **Run length** |
|---|---|
| WAIT | 500 |
| ALLQ | 8000 |
| IDLE | 7000 |

Tables D.2.a, D.2.b. and D.2.c give the c.i. width and c.i. lower and upper limits for different number of replications and different simulation run lengths. We note how once the curve has become horizontal the c.i. covers the real value. Although the results are not shown, for simulation run lengths longer than those given above and for which the curve of the mean estimates becomes horizontal, the c.i. coverage is close to the nominal one of 1 - $\alpha$.

| Results based on 100 replications | | | | | |
|---|---|---|---|---|---|
| Run Length | Mean | Std.Dev. | c.i. half-width | c.i.l.l. | c.i.u.l. |
| 500 | 0.51 | 0.20 | 0.04 | 0.47 | 0.55 |
| 6000 | 0.50 | 0.08 | 0.02 | 0.49 | 0.52 |
| 10000 | 0.51 | 0.04 | 0.01 | 0.50 | 0.52 |
| | Results based on 400 replications | | | | |
| Run Length | Mean | Std.Dev. | c.i. half-width | c.i.l.l. | c.i.u.l. |
| 500 | 0.50 | 0.18 | 0.02 | 0.48 | 0.52 |
| 6000 | 0.51 | 0.08 | 0.01 | 0.50 | 0.52 |
| 10000 | 0.51 | 0.04 | 0.00 | 0.51 | 0.52 |
| | Results based on 700 replications | | | | |
| Run Length | Mean | Std.Dev. | c.i. half-width | c.i.l.l. | c.i.u.l. |
| 500 | 0.50 | 0.18 | 0.01 | 0.49 | 0.52 |
| 6000 | 0.51 | 0.08 | 0.01 | 0.50 | 0.52 |
| 10000 | 0.51 | 0.04 | 0.00 | 0.51 | 0.51 |

**Table D.2.a.** WAIT c.i. width and lower and upper c.i. limits for different simulation run lengths and different number of replications.

| Results based on 100 replications | | | | | |
|---|---|---|---|---|---|
| Run Length | Mean | Std.Dev. | c.i. half-width | c.i.l.l. | c.i.u.l. |
| 500 | 11.86 | 9.87 | 1.96 | 9.90 | 13.82 |
| 6000 | 13.79 | 5.77 | 1.14 | 12.65 | 14.94 |
| 10000 | 14.50 | 3.27 | 0.65 | 13.85 | 15.15 |
| Results based on 400 replications | | | | | |
| Run Length | Mean | Std.Dev. | c.i. half-width | c.i.l.l. | c.i.u.l. |
| 500 | 11.06 | 8.15 | 0.92 | 10.14 | 11.99 |
| 6000 | 13.77 | 5.73 | 0.65 | 13.12 | 14.42 |
| 10000 | 14.58 | 3.36 | 0.38 | 14.20 | 14.96 |
| Results based on 700 replications | | | | | |
| Run Length | Mean | Std.Dev. | c.i. half-width | c.i.l.l. | c.i.u.l. |
| 500 | 10.85 | 8.23 | 0.61 | 10.24 | 11.46 |
| 6000 | 14.09 | 6.45 | 0.48 | 13.61 | 14.57 |
| 10000 | 14.72 | 3.73 | 0.28 | 14.45 | 15.00 |

**Table D.2.b.** ALLQ c.i. width and lower and upper c.i. limits for different simulation run lengths and different number of replications.

| Results based on 100 replications | | | | | |
|---|---|---|---|---|---|
| Run Length | Mean | Std.Dev. | c.i. half-width | c.i.l.l. | c.i.u.l. |
| 500 | 4.03 | 0.68 | 0.13 | 3.89 | 4.16 |
| 6000 | 4.05 | 0.25 | 0.05 | 4.00 | 4.10 |
| 10000 | 4.00 | 0.14 | 0.03 | 3.97 | 4.03 |
| Results based on 400 replications | | | | | |
| Run Length | Mean | Std.Dev. | c.i. half-width | c.i.l.l. | c.i.u.l. |
| 500 | 3.99 | 0.63 | 0.07 | 3.92 | 4.07 |
| 6000 | 4.01 | 0.25 | 0.03 | 3.98 | 4.04 |
| 10000 | 3.99 | 0.14 | 0.02 | 3.98 | 4.01 |
| Results based on 700 replications | | | | | |
| Run Length | Mean | Std.Dev. | c.i. half-width | c.i.l.l. | c.i.u.l. |
| 500 | 4.01 | 0.67 | 0.05 | 3.96 | 4.06 |
| 6000 | 4.00 | 0.26 | 0.02 | 3.98 | 4.02 |
| 10000 | 3.99 | 0.15 | 0.01 | 3.98 | 4.00 |

**Table D.2.c.** IDLE c.i. width and lower and upper c.i. limits for different simulation run lengths and different number of replications.

## D.2.5. THE STEELWORKS MODEL : ESTIMATION OF THE NUMBER OF REPLICATIONS

Two queues have been studied in this model: the TBLOWQ and the PITQ queues, both belonging to the TORPEDO entity. Following the double sequential approach we have estimated a number of 100 replications in order to obtain accurate steady state mean estimates.

## D.2.6. THE STEELWORKS MODEL: EVALUATION OF THE PERFORMANCE OF THE PROPOSED PROCEDURE.

Figures D.7. (TBLOWQ) and D.8. (PITQ) show the mean queuing time estimates as a function of the simulation run length and of the number of replications. They show how the approximation for the number of replications is good and how an increase in the number of replications does not change the shape of the curve.

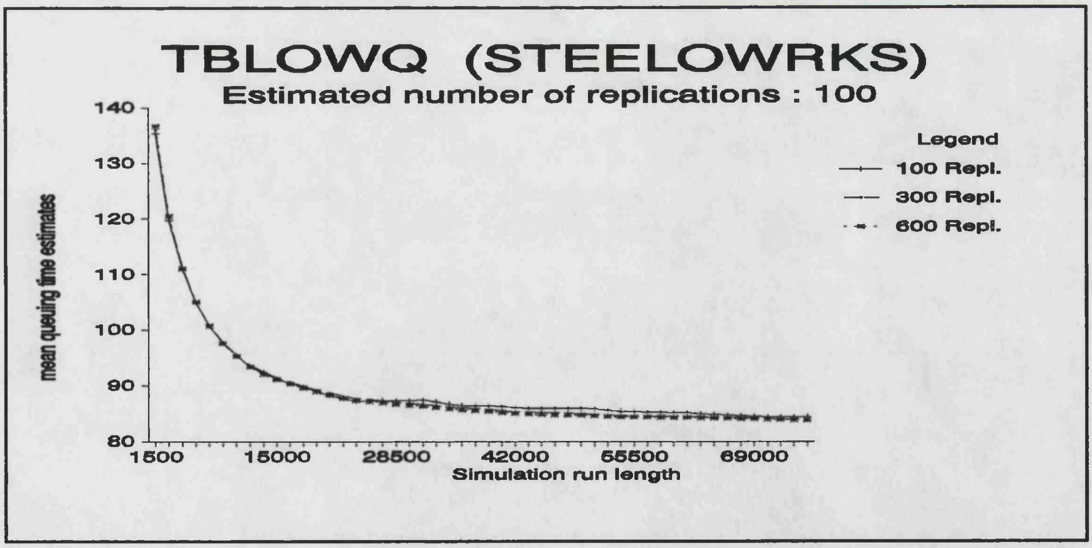



**Figure D.7.** TBLOWQ mean queuing time estimates as a function of the simulation run length and of the number of replications.

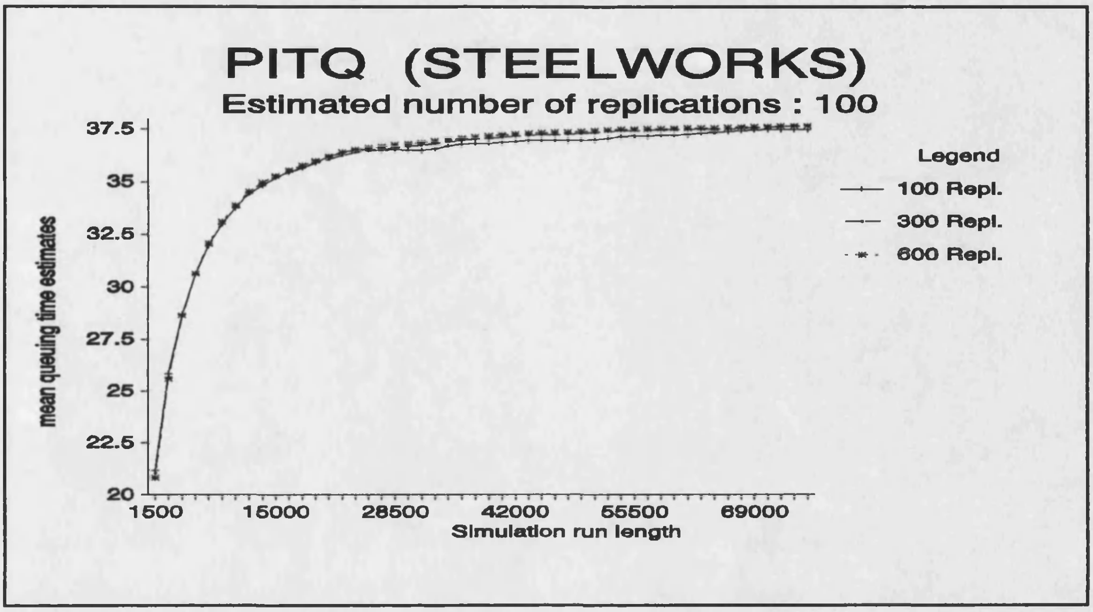



**Figure D.8.** PITQ mean queuing time estimates as a function of the simulation run length and of the number of replications.

# APPENDIX E: GORDON'S METHOD FOR THE ELIMINATION OF THE INFLUENCE OF THE INITIAL CONDITIONS

## E.1. GORDON'S METHOD : DESCRIPTION

This method proposed by Gordon to deal with the initialisation bias problem is a *heuristic* method supported by basic statistical theory. Using the method of replications *k* replications are made. In each replication the sample size is *n*. Therefore k estimates $X_1$, $X_2$, ... $X_k$ are obtained where $X_i$ is the estimate obtained from replication *i*.

The mean estimate $\bar{Y}$ is given by the average of the k different estimates $X_i$:

$$\bar{Y} = 1/k * \sum_{i=1}^{k} X_i$$

and the standard deviation of the mean estimate $\bar{Y}$ is estimated by s:

$$s = \sqrt{\frac{\sum_{i=1}^{k} (X_i - \bar{Y})^2}{(k-1)}}$$

Gordon states that as the individual values $X_i$ are the average of *n* values, then the variance of $X_i$ is proportional to 1/n or, equivalently, the standard deviation will be proportional to $(1/n)^{1/2}$ once the influence of the initial conditions has disappeared. However, due to the high variability of the estimates, it is necessary to obtain *k* replications, and use the average $\bar{Y}$ as the mean estimate for a sample of size *n*. The variance of $\bar{Y}$ is also proportional to 1/n. Therefore the value of n for which a graph of $\mathrm{Log}((\mathrm{Var}(\bar{Y}))^{½})$ vs. Log(n) for increasing values of n becomes linear with a slope of -½ can be chosen as the number of initial observations to be deleted.

## E.2. DISCUSSION OF GORDON'S METHOD

In order to compare this method to the method proposed in Chapter 4 some experiments were performed using it, and once the value "n" of the number of observations to be deleted was approximately estimated, the simulation run length corresponding to this number of observations "n" was determined.

The first problem the simulation practitioner faces is that the determination of the value n for which the graph becomes linear and with slope -½ is not straightforward. By this we mean that in general this slope is not constant but has a variation over a rather large range of values. On the other hand the standard deviation is influenced by the number of replications. And because the number of replications that is required for this influence to disappear depends on the parameter and on the model, the application of this method is not straightforward. This is another reason for which the decrease, at least for a number of replications that is not so large as to make the method of little practical value, does not have a constant slope. All these problems lead to an overestimation of the number of observations to be deleted as is shown in Chapter 4 where the run-in-period obtained using Gordon's method is compared to the run-in-period obtained with our method.

## E.3. EMPIRICAL RESULTS

Run-in-periods were estimated using this last method for the following simulation models:

1. The PUB.
2. The LAUNDERETTE.
3. The STEELWORKS.
4. The M/M/1 queue.
5. The M/M/4 queue.
6. The 2-Stage Queuing System.

For some of the queues of these models discussed in Chapter 4 (or in Appendix F) we show the graph of the logarithm of the standard deviation as a function of the logarithm of the number of observations and try to estimate the value of n for which the graph is linear with slope -½.

## E.3.1. THE PUB

Figures E.1. and E.2 apply Gordon's method to the mean queuing time and the mean queue length of the CLEAN queue of the PUB model. From these figures the graph becomes approximately linear with slope -½ for values of n greater than 500 (i.e. Log 500=2.699). This number of observations corresponds to a simulation run length of 2520. It is important to notice that this value is not exact but an approximation. If the number of observations is fixed, the simulation run length required to take the same number of observations in different replications becomes a random variable; similarly, when we fix the simulation run length the number of observations recorded for this simulation run length in different replications becomes a random variable; this means that in different replications the corresponding random variable (the simulation run length when the number of observations that are recorded is fixed, or vice versa) takes different values, but nevertheless they do not differ greatly.

To show why it is difficult to estimate the number of "customers" for which the standard deviation graph has a slope of -½, Table E.1 gives the values of the standard deviation for the CLEAN mean queuing time as a function of the number of customers, and it also gives the slope of the graph. Data has not been recorded for customers 1, 2... but for groups of L, 2L, 3L... customers. With this convention in mind, the "slope" given in Table E.1. is calculated as :

$$\frac{\text{LOG(Std. Dev}(n+L)) - \text{LOG (Std. Dev. }(n))}{\text{LOG}(n+L) - \text{LOG}(n)}$$

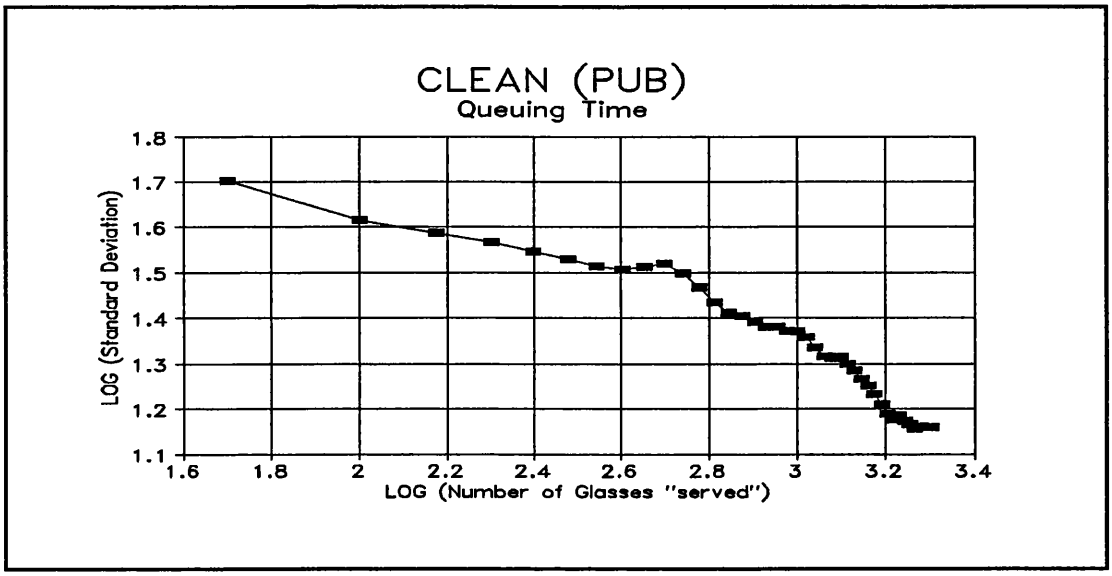



**Figure E.1.** Logarithm of the standard deviation of the mean queuing time as a function of the logarithm of the number of observations in the CLEAN queue.

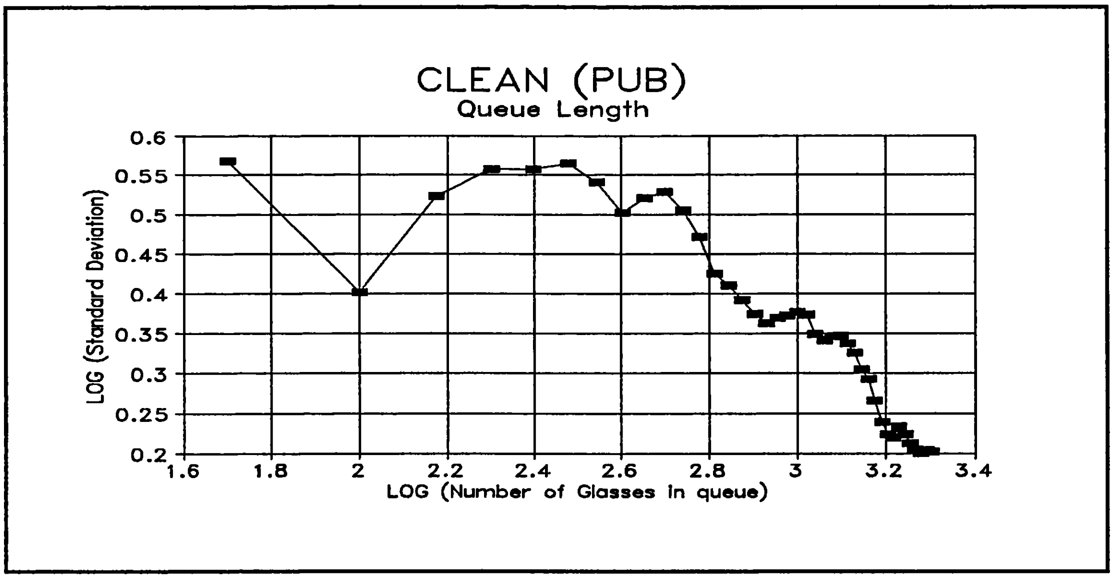



**Figure E.2.** Logarithm of the standard deviation of the mean queue length as a function of the logarithm of the number of observations in the CLEAN queue.

| PUB - GORDON'S RULE | | |
|---|---|---|
| | QUEUING TIME | |
| N.Cust. | Std.Dev. | Slope |
| 50 | 50.488 | |
| 100 | 41.251 | -0.292 |
| 150 | 38.656 | -0.160 |
| 200 | 36.863 | -0.165 |
| 250 | 35.160 | -0.212 |
| 300 | 33.822 | -0.213 |
| 350 | 32.690 | -0.221 |
| 400 | 32.140 | -0.127 |
| 450 | 32.611 | 0.123 |
| 500 | 33.062 | 0.130 |
| 550 | 31.537 | -0.496 |
| 600 | 29.315 | -0.840 |
| 650 | 27.120 | -0.972 |
| 700 | 25.794 | -0.677 |
| 750 | 25.377 | -0.236 |
| 800 | 24.707 | -0.414 |
| 850 | 24.046 | -0.448 |
| 900 | 24.058 | 0.009 |
| 950 | 23.584 | -0.368 |
| 1000 | 23.485 | -0.082 |
| 1050 | 22.816 | -0.592 |
| 1100 | 21.697 | -1.081 |
| 1150 | 20.646 | -1.117 |
| 1200 | 20.497 | -0.171 |
| 1250 | 20.586 | 0.107 |
| 1300 | 19.924 | -0.834 |

**Table E.1.** STANDARD DEVIATION corresponding to the CLEAN mean queuing time as a function of the number of "customers" in the queue.

where "Std. Dev(n+L)" is the standard deviation when "n+l" "customers" have been serviced; if the standard deviation has a constant fall we would expect this value of the slope to be a constant, or at least, close to -½.

From this table it can be noted how the slope is far from being constant and the value of n=500 chosen as the truncation approximation is a very crude approximation.

### E.3.2. THE LAUNDERETTE

Table E.2. gives the same information of Table E.1. for the DRYQ mean queuing time parameter. As in the case of Table E.1. the estimation of the value of *n* for which the slope becomes -½ is not easy and we simply use a very poor and crude approximation for this value.

Figures E.3. and E.4 give the graphs of the logarithms of the standard deviation as a function of the number of observations for the mean queuing time and the mean queue length of the DRYQ. A rough approximation gives a value of n=200 as the value for which the graph becomes approximately linear with a slope of -½. This number of observations corresponds to a simulation run length of 3780.

### E.3.3. THE STEELWORKS

In this system we are interested in obtaining a run-in-period for the mean queuing time and the mean queue length of the TBLOWQ.

The graph for the mean queuing time parameter is given in Figure E.5 and from this graph the number of n for which the graph becomes linear with slope -½ is 220 which roughly corresponds to a simulation run length of 13200. A similar graph for the mean queue length parameter gives an estimated run-in-period of 13200.

| LAUNDERETTE - GORDON'S RULE | | | | | |
|---|---|---|---|---|---|
| No. Cust. | Std. Dev. | Slope | No. Cust. | Std. Dev. | Slope |
| 130 | 9.492 | 0.409 | 470 | 6.227 | -0.769 |
| 140 | 9.711 | 0.308 | 480 | 6.153 | -0.565 |
| 150 | 9.974 | 0.387 | 490 | 6.079 | -0.586 |
| 160 | 10.262 | 0.442 | 500 | 5.976 | -0.847 |
| 170 | 10.519 | 0.409 | 510 | 5.9 | -0.645 |
| 180 | 10.74 | 0.364 | 520 | 5.822 | -0.687 |
| 190 | 10.757 | 0.029 | 530 | 5.714 | -0.983 |
| 200 | 10.639 | -0.215 | 540 | 5.62 | -0.89 |
| 210 | 10.309 | -0.646 | 550 | 5.564 | -0.549 |
| 220 | 9.889 | -0.892 | 560 | 5.539 | -0.245 |
| 230 | 9.512 | -0.875 | 570 | 5.513 | -0.265 |
| 240 | 9.183 | -0.826 | 580 | 5.494 | -0.202 |
| 250 | 8.959 | -0.607 | 590 | 5.504 | 0.111 |
| 260 | 8.778 | -0.519 | 600 | 5.527 | 0.243 |
| 270 | 8.631 | -0.447 | 610 | 5.53 | 0.034 |
| 280 | 8.503 | -0.411 | 620 | 5.526 | -0.049 |
| 290 | 8.409 | -0.317 | 630 | 5.52 | -0.058 |
| 300 | 8.278 | -0.465 | 640 | 5.524 | 0.045 |
| 310 | 8.136 | -0.526 | 650 | 5.552 | 0.324 |
| 320 | 8.004 | -0.516 | 660 | 5.572 | 0.237 |
| 330 | 7.895 | -0.443 | 670 | 5.583 | 0.134 |
| 340 | 7.812 | -0.354 | 680 | 5.571 | -0.145 |
| 350 | 7.787 | -0.114 | 690 | 5.557 | -0.174 |
| 360 | 7.743 | -0.2 | 700 | 5.535 | -0.278 |
| 370 | 7.565 | -0.846 | 710 | 5.499 | -0.461 |
| 380 | 7.389 | -0.885 | 720 | 5.446 | -0.694 |
| 390 | 7.165 | -1.184 | 730 | 5.373 | -0.98 |
| 400 | 6.95 | -1.206 | 740 | 5.302 | -0.975 |
| 410 | 6.831 | -0.7 | 750 | 5.242 | -0.844 |
| 420 | 6.737 | -0.571 | 760 | 5.175 | -0.97 |
| 430 | 6.642 | -0.606 | 770 | 5.111 | -0.964 |
| 440 | 6.561 | -0.534 | 780 | 5.046 | -0.994 |
| 450 | 6.442 | -0.812 | 790 | 5.016 | -0.468 |
| 460 | 6.331 | -0.792 | 800 | 5.01 | -0.089 |

**Table E.2.** STANDARD DEVIATION corresponding to the DRYQ mean queuing time as a function of the number of "customers" in the queue.

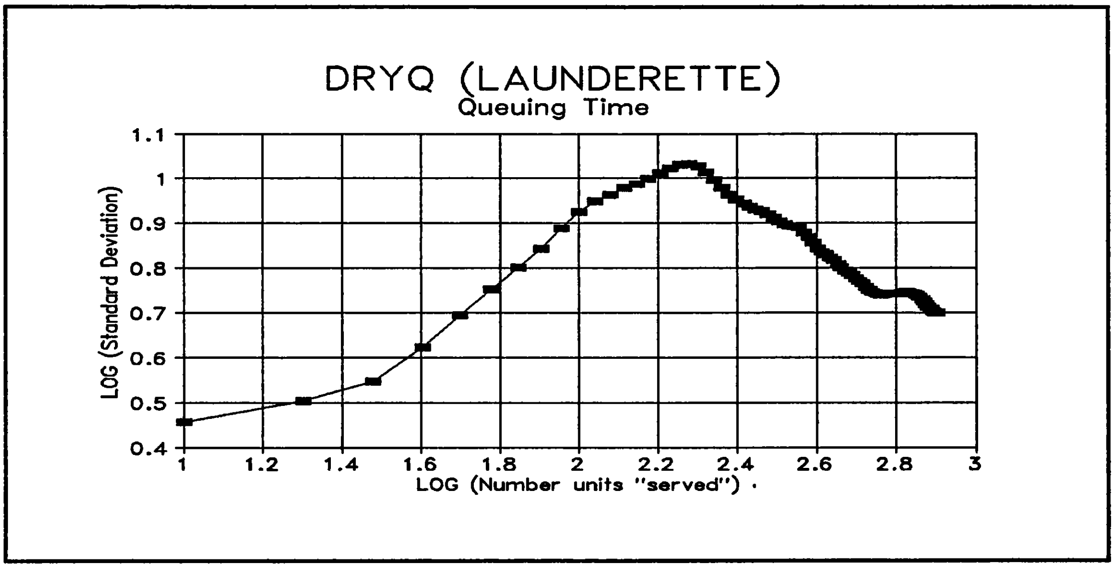



**Figure E.3.** Logarithm of the standard deviation of the mean queuing time as a function of the logarithm of the number of observations in the DRYQ queue.

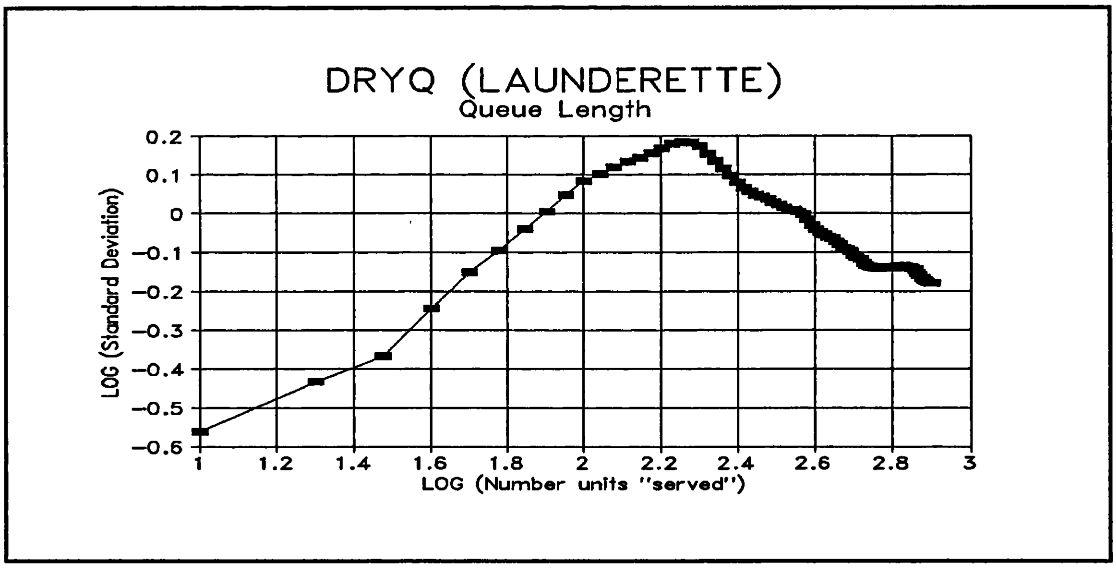



**Figure E.4.** Logarithm of the standard deviation of the mean queuing time as a function of the logarithm of the number of observations in the DRYQ.

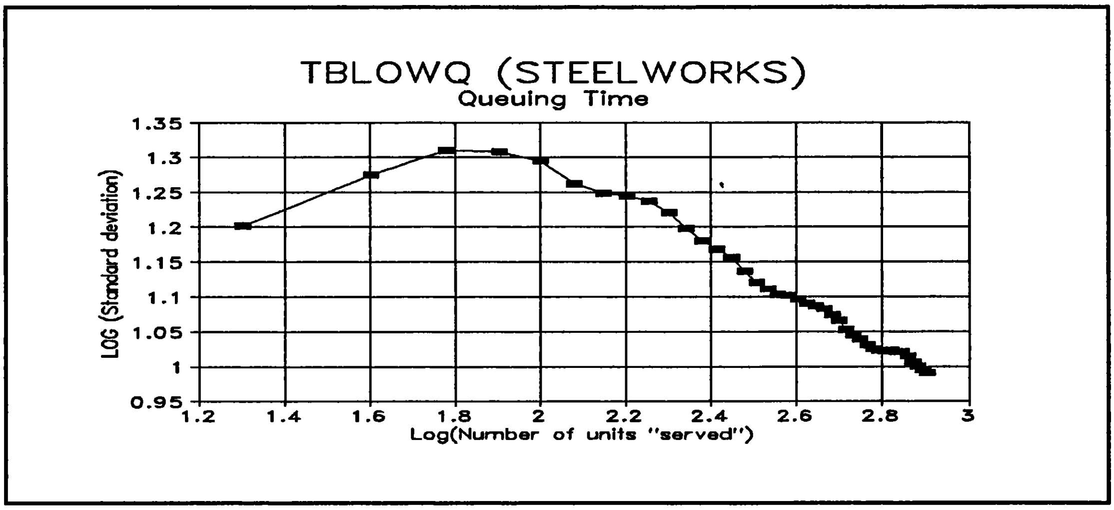



Figure E.5. Logarithm of the standard deviation of the mean queuing time as a function of the logarithm of the number of observations in the TBLOWQ queue.

### E.3.4. THE M/M/1 QUEUE.

The M/M/1 queue with arrival rate $\lambda=1/15$ and service rate $\mu=1/10$ was simulated for different number of "customers" waiting to be served, "n". Figure E.6 gives the graph of the logarithm of the standard deviation of the mean queuing time as a function of the logarithm of n.

The slope of the graph becomes $-\frac{1}{2}$ for approximately Log(n)=2, which corresponds to n=100. The approximate simulation run length is 2500. The graph for the mean queue length parameter is similar and gives the same run-in-period of 2500.

### E.3.5. THE M/M/4 QUEUE

This queue with arrival rate of 1/15 and service rate of 1/50 was simulated for different number of customers being served, and Figure E.7 gives the graph of the logarithm of the standard deviation of the mean queuing time as a function of the logarithm of the number of customers being served.

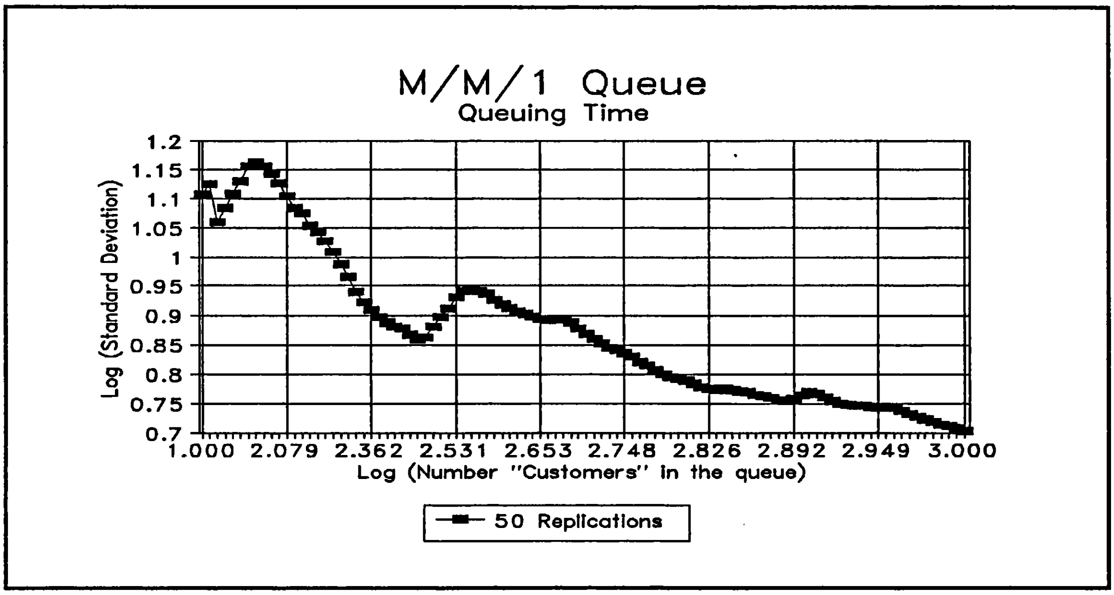



**Figure E.6.** Logarithm of the standard deviation of the mean queueing time as a function of logarithm of the number of observations in the M/M/1 queue.

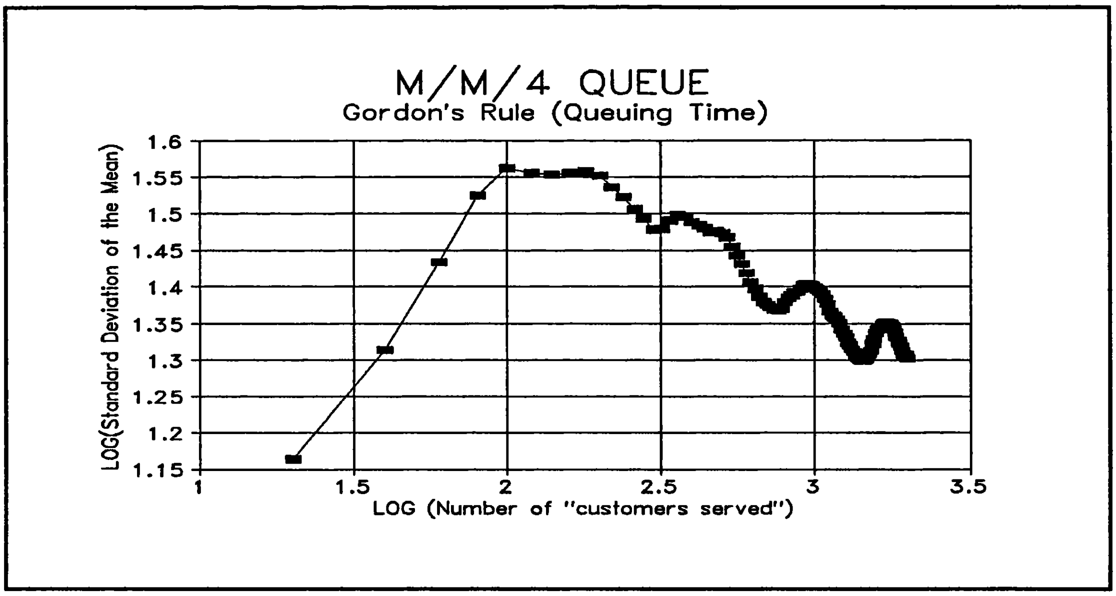



**Figure E.7.** Logarithm of the standard deviation of the mean queuing time as a function of the logarithm of the number of observations in the M/M/4 queue.

From this graph the slope becomes approximately $-\frac{1}{2}$ for values of the X-axis greater or equal to 2.954, which corresponds to 900 customers being served. The corresponding run-in-period can be approximated by 7650.

### E.3.6. JACKSON'S MODEL

The description of this model has been taken from Winston (1987) and was given in Section A.2.7.

The queue for the ENGINES being installed has been studied and the run-in-period for its mean queuing time has been estimated using Gordon's method. Figure E.8. shows the graph of the logarithm of the standard deviation of the mean queuing time as a function of the logarithm of the number of cars in this queue.

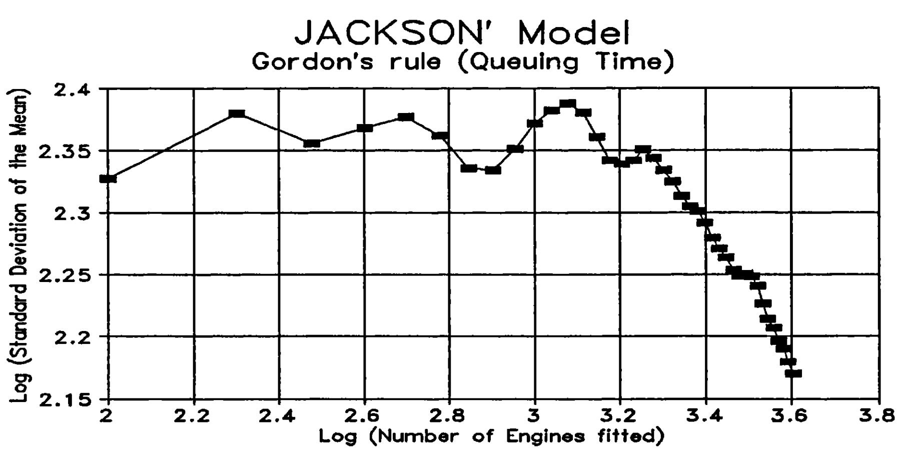



**Figure E.8.** Logarithm of the standard deviation of the mean queuing time as a function of the logarithm of the number of cars waiting for an ENGINE to be fitted.

From the graph we identify a value of 3.279 as the X-axis for which the slope of the curve becomes $-\frac{1}{2}$. This corresponds to a number of 1900 cars having the engine fitted. For this number the approximate run-in-period is 114700.

## APPENDIX F : FURTHER ANALYSIS OF THE EMPIRICAL RESULTS CORRESPONDING TO THE SIMULATION MODELS USED IN CHAPTER 4

### F.1. INTRODUCTION

In this appendix, we analyse some additional results used to test the method based on the standard deviation of the sample mean for the estimation of a suitable run-in-period to reduce the bias due to the influence of the initial conditions.

Results have been obtained for some queues of the LAUNDERETTE, the PUB, and the STEELWORKS models.

In Section F.2. we summarise some of the important points discussed in Chapter 4 concerning the use and evaluation of the method there proposed, and in Section F.3. we discuss the additional results obtained to test the performance of the run-in-periods estimated with our method.

### F.2. MEASURES OF PERFORMANCE.

In this section we define some measures of performance that may be used for the evaluation of the new method proposed here (or any other method) for dealing with the initialisation bias problem.

As was discussed in Chapter 2, a parameter will be considered to be in the steady state when the mean estimates fall within $\epsilon = 2.5\%$ of the real steady state value, $\mu$, calculated in an empirical way in Appendix C.

With this convention in mind, we can define what a "good" run-in-period is. In practice the following three conditions should be met for a run-in-period to deal successfully with the initialisation bias problem:

1. When a run-in-period is used, the observations obtained for short simulation run lengths should be closer to the steady state value $\mu$ as compared to the observations obtained when no run-in-period is used.

2. Estimates that fall within a value $\epsilon$ (see discussion above and in Chapter 2) of the real steady state value should be obtained for shorter simulation run lengths than if no run-in-period or longer or shorter run-in-periods are used.

3. In the case that for more than one value of a run-in-period the parameter reaches the steady state for approximately the same simulation run length, the run-in-period giving estimates closer in absolute value to the steady state value performs better.

## F.3. ANALYSIS OF THE RESULTS

In this section we discuss results obtained for some of the queues of the different simulation models that have been used in this thesis for testing the proposed procedure. The study of these models is divided into two parts: estimation of the run-in-period and evaluation of its performance.

### F.3.1. THE LAUNDERETTE.

Results for the queuing time and the queue length of the DRYQ queue were discussed in Chapter 4. In this appendix we present similar results for the following queues: WASHQ, DRIER, WMIDLE, BIDLE (See Figure A.5, Appendix A).

**1. Estimation of the run-in-periods.**

In order to obtain an estimate of the run-in-period we make 20 replications and obtain the mean estimates for different simulation run lengths, $T_1$, $2T_1$... From empirical results the value of $T_1$ (where $T_1 = T_0/N$, see section 4.4.3), which is not critical, can be chosen to be slightly larger than the maximum activity execution time which in this example has a mean of 40 minutes (Activity LOADW). According to this rule we chose $T_1 = 60$. We also need to calculate the standard deviation of the mean estimates. By either graphing this value as a function of the simulation run length, or by analysing the numerical values, the run-in-period to be used is set equal to the simulation run length for which the standard deviation reaches a maximum. Graphs for the standard deviation corresponding to the mean queuing time of the queues of this model studied in this appendix are given in the following figures:

| FIGURE | QUEUE | PARAMETER |
|---|---|---|
| F.1 | WASHQ | Queuing Time |
| F.2. | BIDLE | Queuing Time |
| F.3. | WMIDLE | Queuing Time |
| F.4. | DRIER | Queuing Time |

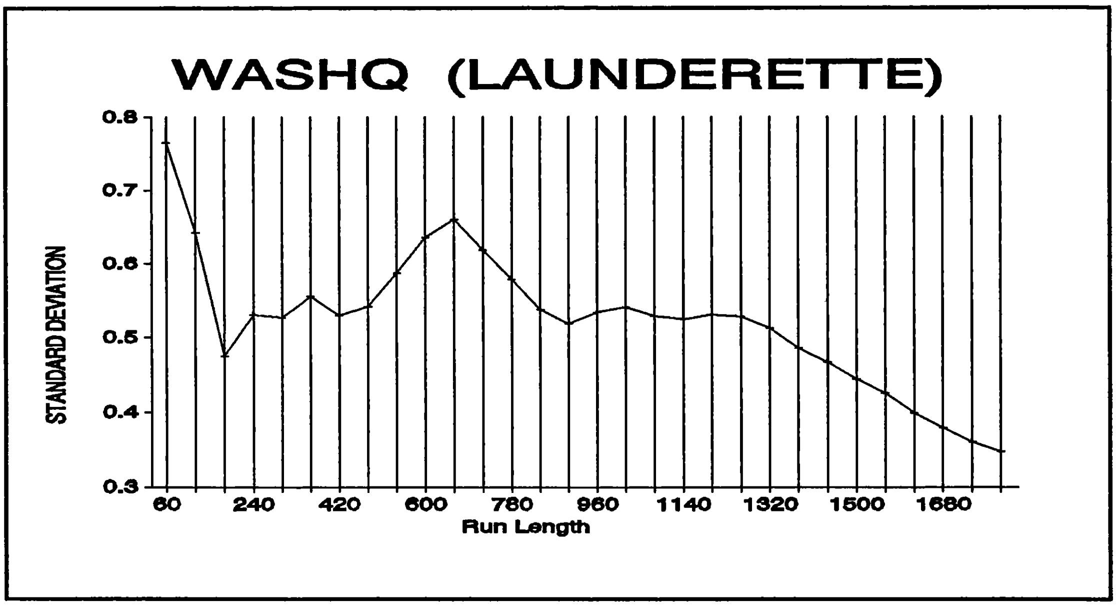



Figure F.1. STANDARD DEVIATION of the WASHQ mean queuing time estimates as a function of the simulation run length.

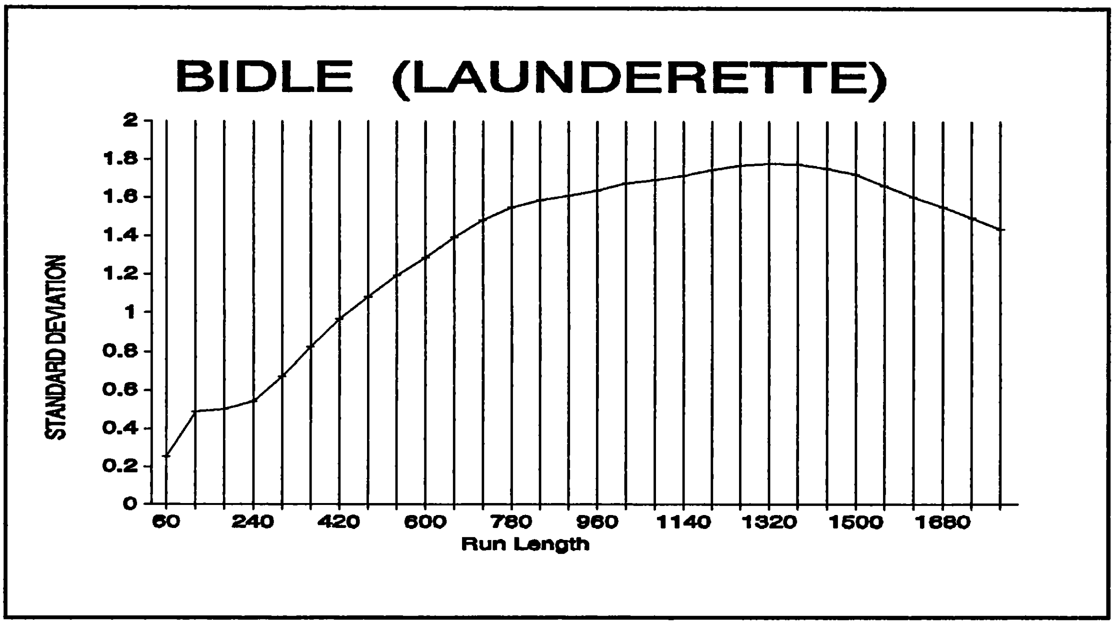



Figure F.2. STANDARD DEVIATION of the BIDLE mean queuing time estimates as a function of the simulation run length.

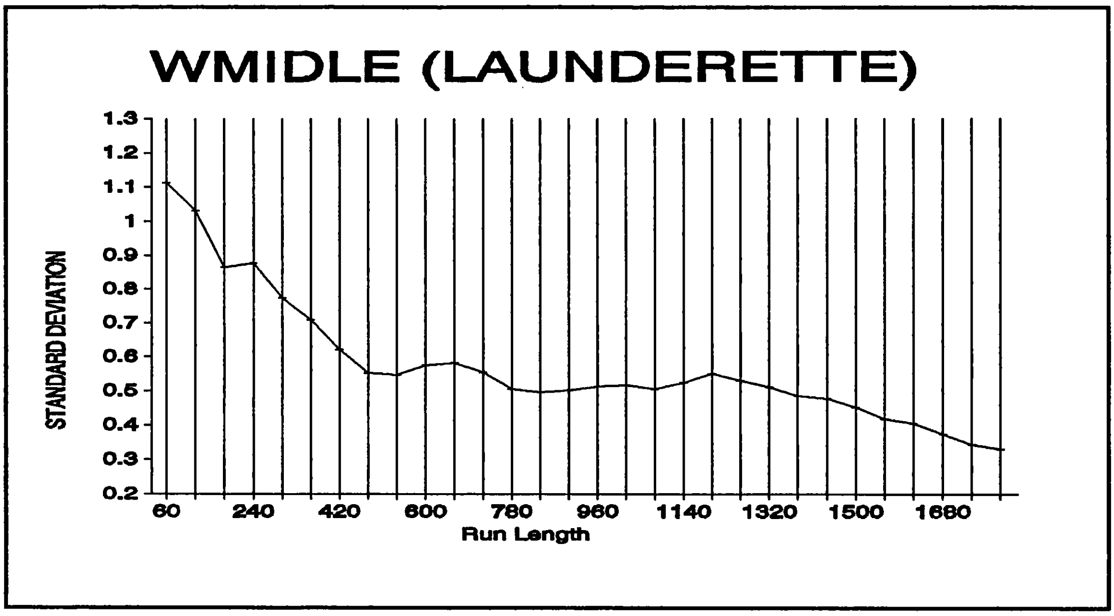



Figure F.3. STANDARD DEVIATION of the WMIDLE mean queuing time estimates as a function of the simulation run length.

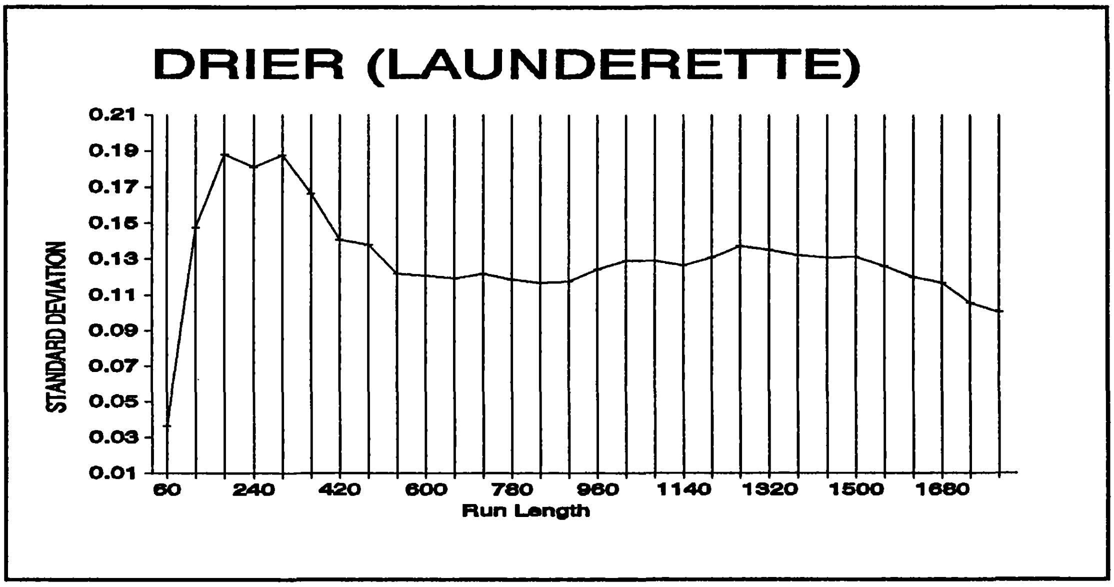



**Figure F.4.** STANDARD DEVIATION of the DRIER mean queuing time estimates as a function of the simulation run length.

From these graphs, or equivalently, from the numerical values for the standard deviation of the mean estimates, given as a function of the simulation run length in Tables F.1a (mean queuing time) and F.1b (mean queue length), the following run-in-periods are obtained:

| QUEUE | PARAMETER | RUN-IN-PERIOD |
|---|---|---|
| WASHQ | Queuing time | 660 |
| WASHQ | Queue length | 660 |
| BIDLE | Queuing time | 1320 |
| BIDLE | Queue length | 1320 |
| WMIDLE | Queuing time | 120 |
| WMIDLE | Queue length | 60 |
| DRIER | Queuing time | 120 |
| DRIER | Queue length | 300 |

**Table F.2.** Run-in-periods estimated for some queues of the LAUNDERETTE model.

| LAUNDERETTE : STANDARD DEVIATION (Mean Queuing Time Estimates) | | | | | |
|---|---|---|---|---|---|
| Run Length | WASHQ | BIDLE | WMIDLE | DRYQ | DRIER |
| 60 | 2.539 | 2.492 | 8.680 | 0.000 | 1.413 |
| 120 | 3.563 | 8.214 | 12.459 | 3.146 | 5.598 |
| 180 | 2.944 | 14.110 | 10.812 | 2.987 | 4.925 |
| 240 | 3.394 | 16.358 | 9.779 | 2.911 | 3.436 |
| 300 | 3.416 | 15.090 | 8.472 | 3.629 | 3.184 |
| 360 | 3.427 | 15.430 | 7.414 | 4.703 | 2.351 |
| 420 | 3.460 | 14.873 | 6.649 | 5.825 | 1.883 |
| 480 | 3.398 | 14.732 | 5.724 | 6.772 | 1.764 |
| 540 | 3.691 | 14.500 | 5.620 | 7.538 | 1.473 |
| 600 | 3.988 | 15.105 | 5.843 | 8.231 | 1.402 |
| 660 | 4.208 | 15.682 | 5.903 | 8.867 | 1.396 |
| 720 | 4.019 | 16.330 | 5.580 | 9.547 | 1.385 |
| 780 | 3.845 | 16.416 | 4.997 | 10.131 | 1.299 |
| 840 | 3.579 | 16.373 | 4.853 | 10.515 | 1.241 |
| 900 | 3.418 | 16.391 | 4.867 | 10.674 | 1.211 |
| 960 | 3.474 | 16.430 | 5.011 | 10.916 | 1.301 |
| 1020 | 3.543 | 16.784 | 5.084 | 11.132 | 1.343 |
| 1080 | 3.514 | 16.962 | 4.843 | 11.344 | 1.331 |
| 1140 | 3.421 | 16.958 | 5.122 | 11.461 | 1.313 |
| 1200 | 3.487 | 17.196 | 5.514 | 11.553 | 1.352 |
| 1260 | 3.542 | 17.877 | 5.445 | 11.758 | 1.434 |
| 1320 | 3.517 | 18.088 | 5.253 | 11.967 | 1.398 |
| 1380 | 3.352 | 17.924 | 4.931 | 12.039 | 1.357 |
| 1440 | 3.229 | 17.648 | 4.927 | 11.919 | 1.338 |

**Table F.1a.** STANDARD DEVIATION corresponding to the mean queuing time estimates as a function of the simulation run length, for different queues of the LAUNDERETTE model.

| LAUNDERETTE : STANDARD DEVIATION (Mean Queue Length Estimates) | | | | | |
|---|---|---|---|---|---|
| Run Length | WASHQ | BIDLE | WMIDLE | DRYQ | DRIER |
| 60 | 0.765 | 0.251 | 1.114 | 0.044 | 0.037 |
| 120 | 0.642 | 0.488 | 1.031 | 0.345 | 0.148 |
| 180 | 0.475 | 0.502 | 0.866 | 0.341 | 0.188 |
| 240 | 0.530 | 0.543 | 0.878 | 0.394 | 0.181 |
| 300 | 0.527 | 0.673 | 0.775 | 0.521 | 0.188 |
| 360 | 0.556 | 0.825 | 0.708 | 0.697 | 0.166 |
| 420 | 0.529 | 0.970 | 0.622 | 0.858 | 0.141 |
| 480 | 0.541 | 1.082 | 0.553 | 0.973 | 0.138 |
| 540 | 0.587 | 1.193 | 0.547 | 1.102 | 0.122 |
| 600 | 0.635 | 1.287 | 0.575 | 1.203 | 0.121 |
| 660 | 0.660 | 1.391 | 0.583 | 1.285 | 0.119 |
| 720 | 0.618 | 1.483 | 0.555 | 1.392 | 0.122 |
| 780 | 0.578 | 1.549 | 0.506 | 1.459 | 0.119 |
| 840 | 0.537 | 1.587 | 0.496 | 1.499 | 0.117 |
| 900 | 0.519 | 1.608 | 0.502 | 1.525 | 0.118 |
| 960 | 0.534 | 1.636 | 0.514 | 1.555 | 0.124 |
| 1020 | 0.541 | 1.676 | 0.517 | 1.588 | 0.129 |
| 1080 | 0.528 | 1.693 | 0.505 | 1.616 | 0.129 |
| 1140 | 0.524 | 1.714 | 0.525 | 1.631 | 0.127 |
| 1200 | 0.531 | 1.745 | 0.550 | 1.657 | 0.131 |
| 1260 | 0.528 | 1.770 | 0.530 | 1.689 | 0.137 |
| 1320 | 0.513 | 1.777 | 0.512 | 1.697 | 0.135 |
| 1380 | 0.486 | 1.774 | 0.486 | 1.694 | 0.132 |
| 1440 | 0.466 | 1.753 | 0.478 | 1.673 | 0.131 |

**Table F.1b.** STANDARD DEVIATION corresponding to the mean queue length estimates as a function of the simulation run length, for different queues of the LAUNDERETTE model.

## 2. Evaluation of the run-in-periods performance

To check in an empirical way how well the estimated run-in-periods perform in terms of the elimination or reduction of the influence of the initial conditions we need to obtain the mean estimates as a function of the simulation run length; to compare the performance of the estimated run-in-period to other run-in-periods, we obtain mean estimates not only for this run-in-period but also for longer and shorter run-in-periods, and when no run-in-period is used. The steady state is reached when the mean estimates fall within 2.5% of the steady state value ($\mu$). The value of $\mu$ (Appendix C), and the range of values for which each parameter can be considered to be in the steady state, for each one of the queues of interest in this model are the following:

| ==== QUEUING TIME VALUES ==== | | |
|---|---|---|
| **Queue** | **Steady state ($\mu$)** | **Range** |
| WASHQ | 6.675 | [6.508 , 6.842] |
| BIDLE | 67.260 | [65.578 , 68.940] |
| WMIDLE | 12.780 | [12.460 , 13.100] |
| DRIER | 1.990 | [1.940 , 2.040] |
| **==== QUEUE LENGTH VALUES ====** | | |
| **Queue** | **Steady state ($\mu$)** | **Range** |
| WASHQ | 0.835 | [0.814 , 0.856] |
| BIDLE | 8.413 | [8.203 , 8.623] |
| WMIDLE | 1.599 | [1.552 , 1.639] |
| DRIER | 0.249 | [0.246 , 0.253] |

The mean queuing time and the mean queue length estimates as a function of the simulation run length and of different run-in-periods are given in tables F.3. (WASHQ), F.4. (BIDLE), F.5. (WMIDLE), and F.6. (DRIER). The columns marked with "**" correspond to the mean estimates obtained with the run-in-period estimated in this appendix, and given in Table F.2.

| WASHQ mean queuing time estimates | | | | | |
|---|---|---|---|---|---|
| Run Length | Run-In 0 | Run-In 120 | Run-In 300 | Run-In 660 | Run-In 1100 |
| | | | | ** | |
| 3000 | 5.976 | 6.124 | 6.183 | 6.203 | 6.163 |
| 4000 | 6.122 | 6.236 | 6.284 | 6.307 | 6.293 |
| 5000 | 6.246 | 6.339 | 6.381 | 6.406 | 6.408 |
| 6000 | 6.361 | 6.440 | 6.478 | 6.509 | 6.514 |
| 7000 | 6.424 | 6.493 | 6.526 | 6.550 | 6.562 |
| 8000 | 6.441 | 6.502 | 6.530 | 6.552 | 6.561 |
| 9000 | 6.488 | 6.543 | 6.569 | 6.590 | 6.600 |
| 10000 | 6.520 | 6.569 | 6.593 | 6.612 | 6.622 |
| 11000 | 6.517 | 6.562 | 6.584 | 6.601 | 6.609 |
| 12000 | 6.533 | 6.575 | 6.595 | 6.610 | 6.618 |

| WASHQ mean queue length estimates | | | | | |
|---|---|---|---|---|---|
| Run Length | Run-In 0 | Run-In 120 | Run-In 300 | Run-In 660 | Run-In 1100 |
| | | | | ** | |
| 3000 | 0.766 | 0.785 | 0.792 | 0.796 | 0.794 |
| 4000 | 0.781 | 0.795 | 0.801 | 0.804 | 0.805 |
| 5000 | 0.794 | 0.806 | 0.811 | 0.814 | 0.816 |
| 6000 | 0.807 | 0.817 | 0.821 | 0.824 | 0.827 |
| 7000 | 0.813 | 0.821 | 0.825 | 0.828 | 0.830 |
| 8000 | 0.814 | 0.821 | 0.825 | 0.827 | 0.829 |
| 9000 | 0.819 | 0.826 | 0.829 | 0.832 | 0.833 |
| 10000 | 0.822 | 0.829 | 0.831 | 0.834 | 0.835 |
| 11000 | 0.821 | 0.827 | 0.829 | 0.831 | 0.833 |

**Table F.3.** WASHQ mean queuing time and mean queue length as a function of the simulation run length and of the run-in-period.

| BIDLE mean queuing time estimates | | | | | |
|---|---|---|---|---|---|
| Run Length | Run-In 0 | Run-In 300 | Run-In 660 | Run-In 1320 | Run-In 1500 |
| | | | | ** | |
| 3000 | 69.885 | 68.687 | 68.280 | 68.2303 | 68.290 |
| 4000 | 69.457 | 68.553 | 68.258 | 68.2504 | 68.303 |
| 5000 | 69.054 | 68.321 | 68.079 | 68.0448 | 68.072 |
| 6000 | 68.797 | 68.181 | 67.979 | 67.9371 | 67.952 |
| 7000 | 68.539 | 68.006 | 67.827 | 67.7739 | 67.782 |
| 8000 | 68.371 | 67.901 | 67.741 | 67.6877 | 67.693 |
| 9000 | 68.177 | 67.756 | 67.610 | 67.5513 | 67.552 |
| 10000 | 68.046 | 67.664 | 67.530 | 67.4718 | 67.470 |
| BIDLE mean queue length estimates | | | | | |
| Run Length | Run-In 0 | Run-In 300 | Run-In 660 | Run-In 1320 | Run-In 1500 |
| | | | | ** | |
| 3000 | 8.660 | 8.502 | 8.451 | 8.424 | 8.420 |
| 4000 | 8.614 | 8.496 | 8.459 | 8.444 | 8.443 |
| 5000 | 8.576 | 8.480 | 8.450 | 8.438 | 8.437 |
| 6000 | 8.551 | 8.471 | 8.446 | 8.436 | 8.435 |
| 7000 | 8.526 | 8.456 | 8.435 | 8.425 | 8.424 |
| 8000 | 8.511 | 8.450 | 8.431 | 8.423 | 8.422 |
| 9000 | 8.494 | 8.439 | 8.422 | 8.414 | 8.413 |
| 10000 | 8.482 | 8.433 | 8.417 | 8.409 | 8.408 |

**Table F.4.** BIDLE mean queuing time and mean queue length estimates as a function of the simulation run length and of the run-in-period.

| WMIDLE queuing time estimates | | | | | |
|---|---|---|---|---|---|
| Run Length | Run-In 0 | Run-In 120 | Run-In 300 | Run-In 660 | Run-In 1100 |
| | | ** | | | |
| 3000 | 13.341 | 13.056 | 13.021 | 13.065 | 13.095 |
| 4000 | 13.255 | 13.041 | 13.016 | 13.047 | 13.067 |
| 5000 | 13.177 | 13.005 | 12.984 | 13.006 | 13.015 |
| 6000 | 13.122 | 12.978 | 12.961 | 12.978 | 12.983 |
| 7000 | 13.059 | 12.935 | 12.919 | 12.932 | 12.932 |
| 8000 | 13.031 | 12.923 | 12.909 | 12.919 | 12.918 |
| 9000 | 12.984 | 12.887 | 12.874 | 12.881 | 12.878 |
| 10000 | 12.958 | 12.871 | 12.859 | 12.865 | 12.862 |
| WMIDLE mean queue length estimates | | | | | |
| Run Length | Run-In 0 | Run-In 60 | Run-In 300 | Run-In 660 | Run-In 1100 |
| | | ** | | | |
| 3000 | 1.657 | 1.611 | 1.609 | 1.610 | 1.608 |
| 4000 | 1.647 | 1.612 | 1.611 | 1.612 | 1.611 |
| 5000 | 1.639 | 1.611 | 1.610 | 1.611 | 1.610 |
| 6000 | 1.633 | 1.610 | 1.609 | 1.610 | 1.609 |
| 7000 | 1.627 | 1.607 | 1.606 | 1.606 | 1.605 |
| 8000 | 1.624 | 1.606 | 1.605 | 1.606 | 1.605 |
| 9000 | 1.619 | 1.604 | 1.603 | 1.603 | 1.602 |
| 10000 | 1.617 | 1.603 | 1.602 | 1.602 | 1.601 |

**Table F.5.** WMIDLE mean queuing time and mean queue length as a function of the simulation run length and of the run-in-period.

| DRIER mean queuing time estimates | | | | | |
|---|---|---|---|---|---|
| Run Length | Run-In 0 | Run-In 120 | Run-In 300 | Run-In 660 | Run-In 1100 |
| | | ** | | | |
| 3000 | 2.432 | 2.105 | 2.063 | 2.053 | 2.057 |
| 4000 | 2.337 | 2.093 | 2.062 | 2.056 | 2.059 |
| 5000 | 2.269 | 2.074 | 2.049 | 2.043 | 2.044 |
| 6000 | 2.228 | 2.066 | 2.045 | 2.040 | 2.041 |
| 7000 | 2.193 | 2.055 | 2.037 | 2.032 | 2.032 |
| 8000 | 2.166 | 2.040 | 2.029 | 2.025 | 2.024 |
| 9000 | 2.141 | 2.034 | 2.019 | 2.015 | 2.014 |
| 10000 | 2.125 | 2.028 | 2.015 | 2.011 | 2.010 |
| 11000 | 2.110 | 2.022 | 2.010 | 2.006 | 2.005 |
| 12000 | 2.098 | 2.017 | 2.006 | 2.003 | 2.002 |
| 13000 | 2.084 | 2.010 | 2.000 | 1.996 | 1.995 |

**Table F.6.** DRIER mean queuing time estimates as a function of the simulation run length and of the run-in-period.

We have not given the mean queue length estimates for the DRIER queue in Table F.6. as the mean steady state value is very small and as discussed in Section 2.6.1. we consider that in these cases a different estimator from the mean should be used. We have underlined in the previous tables the mean estimates beyond which the parameter can be considered to be in the steady state. From these values we can notice how in all the cases, except for the DRIER mean queuing time estimates, the steady state is reached for shorter simulation run lengths when the estimated run-in-period is used. In the case of the DRIER queue if a run-in-period of 660 is used, the steady state is reached for a simulation run length of 6000, instead of a simulation run length of 7000. The difference is not big enough as to make the proposed method useless, and considering that we deal with stochastic systems, there is always a small margin of uncertainty which is why we do not expect optimal, but good results. But nevertheless, the mean estimates in the columns marked "**" are usually closer to the value of $\mu$, especially when the steady state is also reached for the same simulation run length using a shorter run-in-period (for example Table F.4, BIDLE queue).

### F.3.2. THE PUB.

#### 1. Estimation of the run-in-periods

Run-in-periods are estimated for the mean queuing time and the mean queue length of three queues of this model: the WAIT, the CLEAN and the IDLE queue. Table F.7. gives the different probability distributions of the different execution times of the activities of this model:

| ACTIVITY | EXECUTION TIME (Probability Distribution) |
|---|---|
| ARRIVE | NEGATIVE EXPONENTIAL , MEAN 15; |
| POUR | NORMAL; MEAN : 6; STANDARD DEVIATION:3; |
| DRINK | UNIFORM between 5 and 9; |
| WASH | 5 |

**Table F.7.** Probability distributions of the execution time of the different activities of the PUB model.

Considering the mean value of the probability distributions which appear in Table F.7. the maximum activity execution time takes 15 units of time. To show that the selection of the value $T_1$ is not critical as long as it is not extremely long, a value of $T_1=60$ was used (4 times greater than the maximum activity execution time). As in previous examples, we graph the standard deviation estimates corresponding to the mean queuing time parameters as a function of the simulation run length, for periods of time of 60, 120... minutes. These graphs are given in Figures F.5. (CLEAN), F.6. (IDLE), and F.7. (WAIT). Similar graphs can be drawn for the mean queue length parameters.

From these graphs (and similar ones for the mean queue length parameters) we estimate the following run-in-periods:

| Queue | Parameter | Estimated Run-In-Period |
|---|---|---|
| WAIT | Queuing time | 120 |
| IDLE | Queuing time | 60 |
| CLEAN | Queuing time | 540 |
| WAIT | Queue length | 120 |
| IDLE | Queue length | 60 |
| CLEAN | Queue length | 180 |

**Table F.8.** Estimated run-in-periods for the queues of interest in the PUB model.

**2. Evaluation of the run-in-periods performance.**

To evaluate the behaviour of the run-in-periods estimated with our method we obtained the mean estimates as a function of the simulation run length and for different run-in-periods.

The steady state as well as the range of values that fall within 2.5% of the value of $\mu$ given in Appendix C, and for which the corresponding parameter can be considered to be in the steady state, are the following:

===== QUEUING TIME VALUES =====

| Queue | Steady state ($\mu$) | Range |
|---|---|---|
| WAIT | 1.141 | [1.112 , 1.169] |
| CLEAN | 209.400 | [204.1 , 214.6] |
| IDLE | 2.001 | [1.950 , 2.05 ] |

===== QUEUE LENGTH VALUES =====

| Queue | Steady state ($\mu$) | Range |
|---|---|---|
| WAIT | 0.228 | [0.222 , 0.233] |
| CLEAN | 41.880 | [40.830 , 42.930] |
| IDLE | 0.800 | [0.780 , 0.820] |

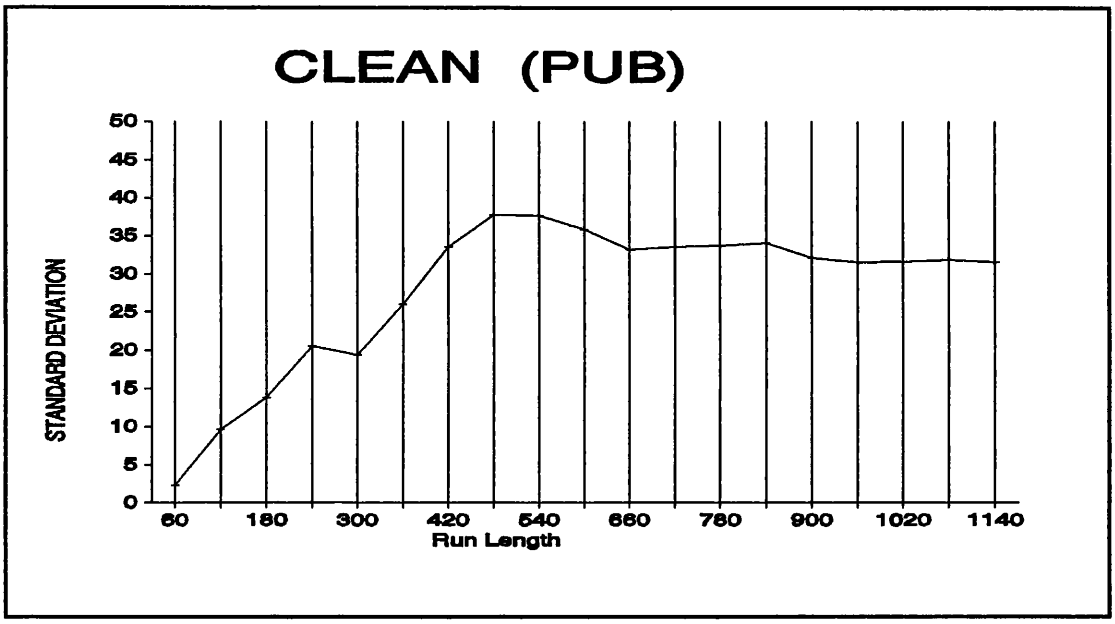



Figure F.5. STANDARD DEVIATION of the CLEAN mean queuing time estimates as a function of the simulation run length.

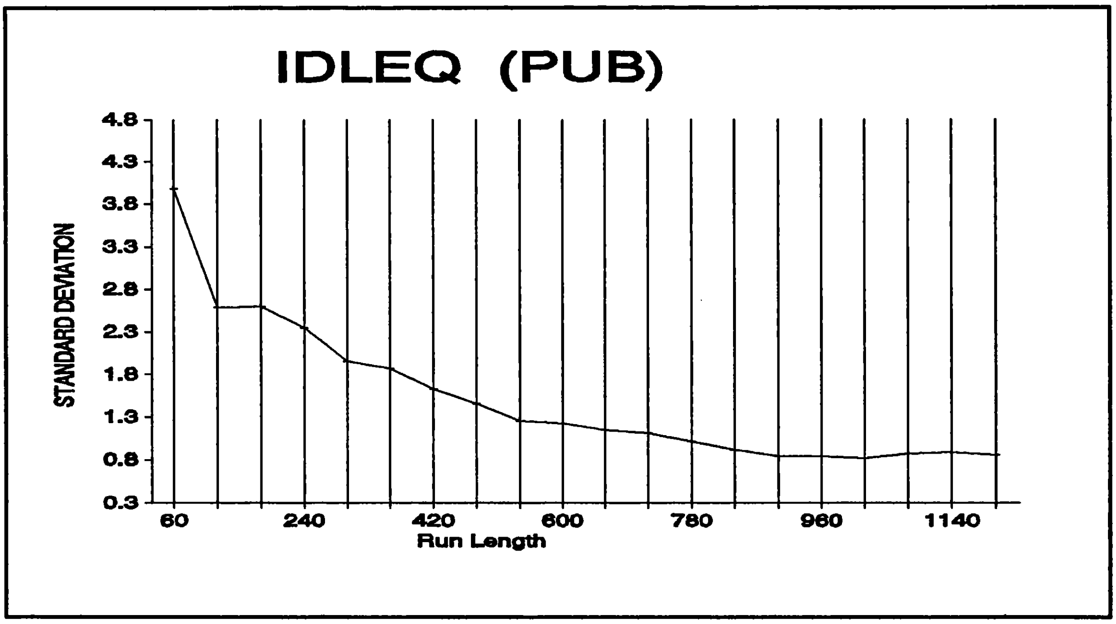



Figure F.6. STANDARD DEVIATION of the IDLE mean queuing time estimates as a function of the simulation run length.

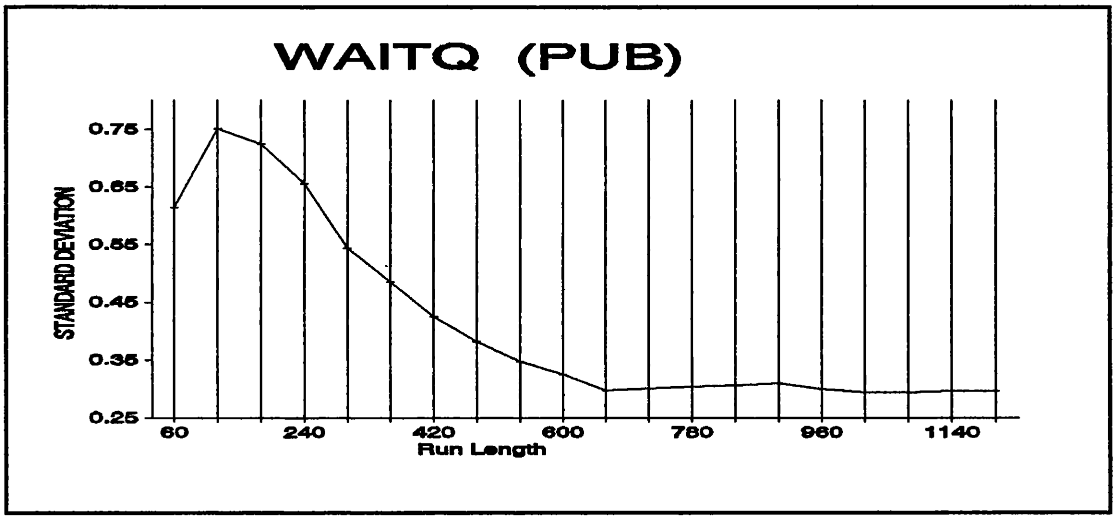



Figure F.7. STANDARD DEVIATION of the WAIT mean queuing time estimates as a function of the simulation run length.

Tables F.9., F.10. and F.11. give the mean queuing time and the mean queue length estimates for the WAIT, CLEAN, and IDLE queues respectively. Columns marked "**" give the mean estimates obtained using the estimated run-in-period (See Table F.8). The underlined values correspond to the estimates for which the parameter can be considered to be in the steady state, i.e., for the simulation run length corresponding to the underlined value, as well as for longer run lengths, the mean estimates fall within 2.5% of the steady state value.

From the values in Table F.9. for the WAIT mean queuing time estimates, we notice that the mean estimates are closer to $\mu$ when a run-in-period of 120, estimated with our method, is used. We have not given the results for the mean queue length estimates because of the small steady state value. As discussed in Section 2.6.1, from a statistical point of view, a different estimator should be used. However, although the results are not included in this thesis, when the run-in-period given in Table F.8. was applied to the queue length parameter, the steady state was reached for a shorter simulation run length than if no run-in-period or longer run-in-periods had been used. This confirms the robustness of our method.

| WAIT mean queuing time estimates | | | | |
|---|---|---|---|---|
| Run Length | Run-in 0 | Run-In 60 | Run-In 120 | Run-In 300 |
| | | | ** | |
| 500 | 0.894 | 0.939 | 0.947 | 0.948 |
| 1000 | 1.018 | 1.044 | 1.052 | 1.071 |
| 1500 | 1.059 | 1.077 | 1.083 | 1.098 |
| 2000 | 1.095 | 1.109 | 1.114 | 1.128 |
| 2500 | 1.109 | 1.120 | 1.124 | 1.135 |
| 3000 | 1.109 | 1.119 | 1.123 | 1.132 |
| 3500 | 1.115 | 1.124 | 1.127 | 1.135 |
| 4000 | 1.103 | 1.111 | 1.114 | 1.120 |
| 4500 | 1.105 | 1.111 | 1.114 | 1.119 |
| 5000 | 1.108 | 1.114 | 1.117 | 1.121 |
| 5500 | 1.106 | 1.111 | 1.113 | 1.118 |
| 6000 | 1.108 | 1.113 | 1.115 | 1.119 |

**Table F.9.** WAIT mean queuing time estimates as a function of the simulation run length and of the run-in-period.

From the values in Table F.10. for the CLEAN mean queuing time estimates we notice that a run-in-period of 120 would give estimates closer to $\mu$, and that would reach the steady state for shorter run lengths than those obtained with the run-in-period estimated with our method. However, the difference between a simulation run length of 500 and a simulation run length of 1500 is not extremely large, and we have to consider that there is always in simulation some uncertainty and randomness in the results we do not expect the procedure to give perfect results in every single case where it is applied. And at the same time we have emphasised that we are looking for **good** and not for **optimal** run-in-periods. In Appendix E we estimated a run-in-period of 2520 for the parameters of the CLEAN queue using Gordon's method. From the underlined values, it is clear that the steady state is reached for simulation run lengths shorter than this value. Similarly, using Conway's method the run-in-period is estimated as 200, which in this case gives a better approximation than the one determined with our method.

| CLEAN mean queuing time estimates | | | | |
|---|---|---|---|---|
| Run Length | Run-in 0 | Run-In 120 | Run-In 300 | Run-In 540 |
| | | | | ** |
| 500 | 179.302 | 210.024 | 224.920 | |
| 1000 | 195.265 | 210.574 | 216.238 | 214.800 |
| 1500 | 200.475 | 210.587 | 214.027 | 213.359 |
| 2000 | 202.498 | 210.041 | 212.497 | 212.028 |
| 2500 | 204.161 | 210.177 | 212.121 | 211.777 |
| 3000 | 205.005 | 209.988 | 211.556 | 211.213 |
| 3500 | 205.260 | 209.504 | 210.796 | 210.436 |
| 4000 | 206.076 | 209.801 | 210.940 | 210.637 |
| 4500 | 206.615 | 209.924 | 210.932 | 210.664 |
| 5000 | 206.809 | 209.780 | 210.677 | 210.426 |
| 5500 | 207.334 | 210.044 | 210.864 | 210.647 |
| 6000 | 207.943 | 210.434 | 211.197 | 211.021 |
| CLEAN mean queue length estimates | | | | |
| Run Length | Run-in 0 | Run-In 120 | Run-In 180 | Run-In 300 |
| | | | ** | |
| 500 | 43.337 | 42.431 | 42.245 | 41.927 |
| 1000 | 42.487 | 41.981 | 41.875 | 41.720 |
| 1500 | 42.337 | 42.001 | 41.936 | 41.852 |
| 2000 | 42.154 | 41.896 | 41.845 | 41.779 |
| 2500 | 42.079 | 41.872 | 41.831 | 41.780 |
| 3000 | 42.054 | 41.882 | 41.849 | 41.808 |
| 3500 | 41.998 | 41.849 | 41.821 | 41.785 |
| 4000 | 42.040 | 41.912 | 41.888 | 41.859 |
| 4500 | 42.046 | 41.932 | 41.911 | 41.887 |
| 5000 | 42.030 | 41.927 | 41.908 | 41.886 |
| 5500 | 42.050 | 41.958 | 41.941 | 41.922 |
| 6000 | 42.061 | 41.977 | 41.961 | 41.944 |
| 6500 | 42.060 | 41.982 | 41.969 | 41.953 |

**Table F.10.** CLEAN mean queuing time and mean queue length estimates as a function of the simulation run length and of the run-in-period.

| IDLE mean queuing time estimates | | | | |
|---|---|---|---|---|
| Run Length | Run-in 0 | Run-In 60 | Run-In 120 | Run-In 300 |
| | | ** | | |
| 500 | 2.839 | 2.520 | 2.450 | 2.608 |
| 1000 | 2.370 | 2.211 | 2.164 | 2.145 |
| 1500 | 2.241 | 2.135 | 2.102 | 2.084 |
| 2000 | 2.180 | 2.101 | 2.075 | 2.063 |
| 2500 | 2.151 | 2.088 | 2.068 | 2.058 |
| 3000 | 2.120 | 2.067 | 2.050 | 2.040 |
| 3500 | 2.093 | 2.048 | 2.033 | 2.024 |
| 4000 | 2.088 | 2.048 | 2.035 | 2.028 |
| 4500 | 2.074 | 2.039 | 2.028 | 2.020 |
| 5000 | 2.066 | 2.035 | 2.024 | 2.018 |
| 5500 | 2.066 | 2.037 | 2.028 | 2.022 |
| 6000 | 2.069 | 2.043 | 2.034 | 2.029 |
| 6500 | 2.063 | 2.039 | 2.031 | 2.026 |
| 7000 | 2.058 | 2.035 | 2.028 | 2.023 |
| 7500 | 2.059 | 2.038 | 2.031 | 2.027 |
| 8000 | 2.054 | 2.034 | 2.028 | 2.024 |
| 8500 | 2.055 | 2.036 | 2.030 | 2.027 |
| 9000 | 2.050 | 2.032 | 2.026 | 2.023 |
| 9500 | 2.049 | 2.033 | 2.027 | 2.024 |
| IDLE mean queue length estimates | | | | |
| Run Length | Run-in 0 | Run-In 60 | Run-In 120 | Run-In 300 |
| | | ** | | |
| 500 | 0.956 | 0.849 | 0.820 | 0.804 |
| 1000 | 0.868 | 0.813 | 0.800 | 0.787 |
| 1500 | 0.847 | 0.810 | 0.801 | 0.794 |
| 2000 | 0.835 | 0.807 | 0.800 | 0.795 |
| 2500 | 0.830 | 0.808 | 0.803 | 0.799 |
| 3000 | 0.823 | 0.805 | 0.800 | 0.797 |
| 3500 | 0.817 | 0.801 | 0.798 | 0.795 |
| 4000 | 0.817 | 0.803 | 0.800 | 0.797 |
| 4500 | 0.814 | 0.801 | 0.799 | 0.797 |
| 5000 | 0.812 | 0.801 | 0.799 | 0.797 |
| 5500 | 0.813 | 0.803 | 0.801 | 0.799 |

**Table F.11.** IDLE mean queuing time and mean queue length estimates as a function of the simulation run length and of the run-in-period.

### F.3.3. THE STEELWORKS

#### 1. Estimation of the run-in-periods

Three queues are studied in this model: TBLOWQ, PITQ, and LOADQ. Table F.12. gives the different execution times of the activities of the system:

| Activity | Execution time (Probability Distribution) |
|---|---|
| MELT | NORMAL; MEAN : 110; Standard Deviation: 15; |
| BLOW | 10; |
| GOING | NEGATIVE EXPONENTIAL; MEAN: 10; |
| FILL | 10 |
| LOADIN | 10; |
| REFINE | 50 + NEGATIVE EXPONENTIAL, MEAN: 50; |
| TRAVEL | 2; |
| RETURN | 4; |

**Table F.12.** Probability distributions of the execution time of the different activities in the STEELWORKS model.

Looking at the different execution times of the activities of this model, a value of $T_1 = 500$ was chosen. The standard deviation of the mean estimates as a function of the simulation run length were obtained and are given in Table F.13. for different simulation run lengths, for each one of the different queues (except the TBLOWQ queue), and for both queue parameters: queuing time and queue length. However, because the graphical approach is easier to analyse, we show in Figure F.8. the graph of the standard deviation of the TBLOWQ mean queuing time as a function of the simulation run length.

| STANDARD DEVIATION (STEELWORKS) | | | | |
|---|---|---|---|---|
| Run Length | PITQ | PITQ | LOADQ | LOADQ |
| | Queuing Time | Queue Length | Queuing Time | Queue Length |
| 500 | 2.954 | 0.254 | 7.634 | 0.182 |
| 1000 | 3.913 | 0.362 | 4.431 | 0.156 |
| 1500 | 5.009 | 0.463 | 2.666 | 0.114 |
| 2000 | 5.074 | 0.471 | 2.037 | 0.095 |
| 2500 | 5.114 | 0.490 | 1.776 | 0.085 |
| 3000 | 5.477 | 0.532 | 1.522 | 0.075 |
| 3500 | 6.043 | 0.581 | 1.460 | 0.072 |
| 4000 | 6.420 | 0.607 | 1.364 | 0.067 |
| 4500 | 6.565 | 0.615 | 1.162 | 0.059 |
| 5000 | 6.406 | 0.600 | 1.055 | 0.054 |
| 5500 | 6.175 | 0.579 | 0.955 | 0.050 |
| 6000 | 5.957 | 0.564 | 0.827 | 0.042 |
| 6500 | 5.847 | 0.552 | 0.702 | 0.035 |
| 7000 | 5.660 | 0.532 | 0.605 | 0.031 |
| 7500 | 5.588 | 0.528 | 0.562 | 0.028 |
| 8000 | 5.491 | 0.518 | 0.556 | 0.028 |

**Table F.13.** Standard deviation of the mean queuing time and the mean queue length estimates as a function of the simulation run length, for some of the queues of the STEELWORKS model.

Using the results of Table F.13. or of graphs similar to the one shown in Figure F.8 we obtained the following run-in-periods, corresponding to the simulated time for which the STANDARD DEVIATION reaches its maximum:

| QUEUE | PARAMETER | Estimated Run-In-Period |
|---|---|---|
| TBLOWQ | Queuing Time | 6500 |
| TBLOWQ | Queue Length | 6500 |
| PITQ | Queuing Time | 4500 |
| PITQ | Queue Length | 4500 |
| LOADQ | Queuing Time | 500 |
| LOADQ | Queue Length | 500 |

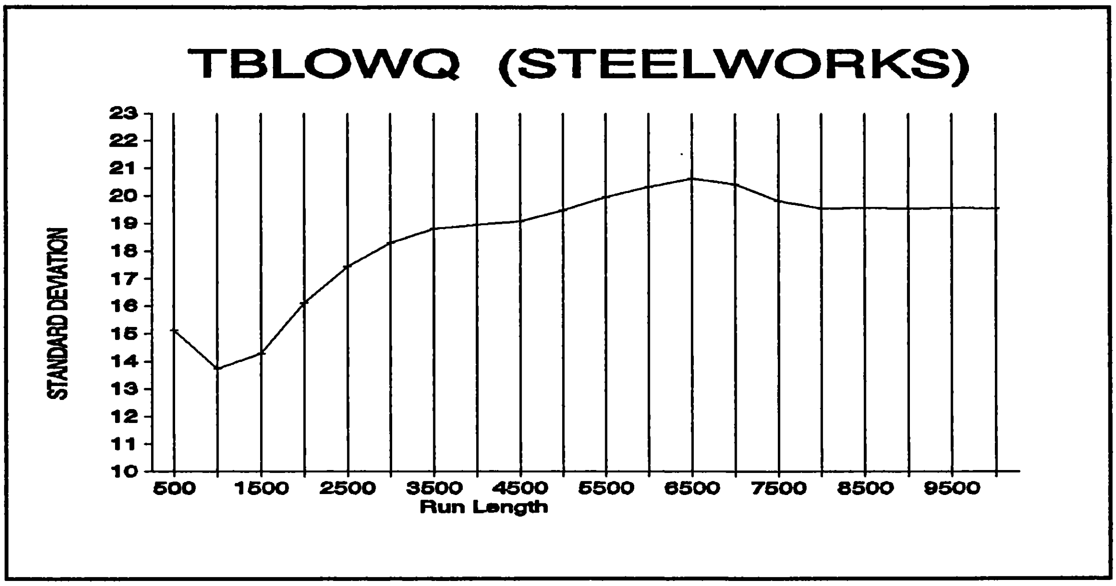



**Figure F.8.** STANDARD DEVIATION of the TBLOWQ mean queuing time estimates as a function of the simulation run length.

## 2. Evaluation of the run-in-periods performance

To evaluate the performance of the run-in-periods estimated using the proposed method, we obtained the mean estimates as a function of the simulation run length for different run-in-periods, including those estimated above, and compared them to the real steady state value ($\mu$).

The values of $\mu$ as well as the range for which the mean estimates can be considered to be in the steady state (2.5% around $\mu$) are the following:

=== QUEUING TIME VALUES ===

| Queue | Steady state ($\mu$) | Range |
|---|---|---|
| TBLOWQ | 81.588 | [ 79.548 , 83.628] |
| PITQ | 38.404 | [37.444 , 39.364] |
| LOADQ | 11.521 | [11.233 , 11.809] |

=== QUEUE LENGTH VALUES ===

| **Queue** | **Steady state ($\mu$)** | **Range** |
|---|---|---|
| TBLOWQ | 2.674 | [2.607 , 2.741] |
| PITQ | 3.602 | [3.512 , 3.692] |
| LOADQ | 0.707 | [0.689 , 0.725] |

Tables F.14., F.15., and F.16. give the mean queuing time and the mean queue length for the TBLOWQ, PITQ, and LOADQ respectively (for this queue we only give the mean queuing time estimates), as a function of the simulation run length and of the run-in-periods. The underlined values correspond to those for which the parameter can be considered to be in the steady state.

The run-in-period (Appendix E) for the TBLOWQ using Gordon's method is estimated as 12380. From the values in Table F.14. we can see that as in most of the examples already discussed, the steady state is reached for a shorter or at least for the same simulation run length if the run-in-period estimated with our method than if the run-in period estimated with Gordon's method is used. In this case, for both the mean queuing time and the mean queue length parameters the steady state is reached for the same simulation run length, but the estimate of the standard deviation is greater when a longer run-in-period is used (i.e., Gordon's run-in-period). The run-in-period estimated Conway's method is 500 which in this case is underestimated.

From the values in Table F.15. for the **PITQ queuing time** parameter we observe that when the run-in-period estimated in this appendix is used the parameter reaches the steady state for a simulation run length of 10000 as compared to a run length of 50000 that is required when no run-in-period is used. At the same time, at least as compared to the other run-in-periods for which estimates were obtained in this experiment, the run-in-period for which convergence occurs earlier in simulated time is the one estimated with our method. Results for the queue length of this queue are similar, and the steady state is reached for the same simulation run lengths as the mean queuing time.

| STEELWORKS | | | | |
|---|---|---|---|---|
| TBLOWQ MEAN queuing time estimates | | | | |
| Run Length | Run-In 0 | Run-In 2500 | Run-In 6500 | Run-In 13200 |
| | | | ** | (Gordon's) |
| 15000 | 92.388 | 85.947 | 82.328 | 81.413 |
| 20000 | 89.904 | 84.968 | 81.529 | 82.078 |
| 25000 | 88.126 | 84.059 | 82.514 | 81.648 |
| 30000 | 87.119 | 83.706 | 81.235 | 81.768 |
| 35000 | 86.500 | 83.553 | 81.390 | 81.990 |
| 40000 | 85.892 | 83.303 | 81.490 | 81.921 |
| 45000 | 85.480 | 83.160 | 81.356 | 81.961 |
| 50000 | 84.858 | 82.750 | 81.354 | 81.598 |
| 55000 | 84.685 | 82.728 | 81.326 | 81.746 |
| 60000 | 84.337 | 82.528 | 81.317 | 81.627 |
| 65000 | 84.065 | 82.388 | 8.130 | 81.549 |
| 70000 | 83.832 | 82.273 | 81.291 | 81.487 |
| TBLOWQ mean queue length estimates | | | | |
| Run Length | Run-In 0 | Run-In 2500 | Run-In 6500 | Run-In 13200 |
| 15000 | 3.039 | 2.818 | 2.697 | 2.666 |
| 20000 | 2.954 | 2.785 | 2.670 | 2.689 |
| 25000 | 2.894 | 2.755 | 2.672 | 2.676 |
| 30000 | 2.860 | 2.743 | 2.662 | 2.679 |
| 35000 | 2.838 | 2.737 | 2.668 | 2.686 |
| 40000 | 2.818 | 2.729 | 2.672 | 2.684 |
| 45000 | 2.805 | 2.726 | 2.667 | 2.687 |
| 50000 | 2.785 | 2.712 | 2.666 | 2.675 |
| 55000 | 2.778 | 2.711 | 2.665 | 2.679 |
| 60000 | 2.766 | 2.705 | 2.665 | 2.675 |
| 65000 | 2.758 | 2.700 | 2.658 | 2.673 |
| 70000 | 2.750 | 2.696 | 2.664 | 2.671 |

**Table F.14.** Mean TBLOWQ queuing time estimates as a function of the simulation run length and of the run-in-period.

| === PITQ mean queuing time estimates === | | | | |
|---|---|---|---|---|
| Run Length | Run-In 0 | Run-In 500 | Run-In 2500 | Run-In 4500 |
| | | | | ** |
| 5000 | 29.382 | 30.540 | 33.960 | 35.162 |
| 10000 | 33.614 | 34.323 | 36.467 | **37.470** |
| 15000 | 35.225 | 35.730 | 37.233 | 37.910 |
| 20000 | 36.030 | 36.420 | **37.579** | 38.072 |
| 25000 | 36.485 | 36.798 | 37.747 | 38.125 |
| 30000 | 36.786 | 37.051 | 37.841 | 38.156 |
| 35000 | 37.027 | 37.256 | 37.933 | 38.206 |
| 40000 | 37.254 | **37.456** | 38.051 | 38.295 |
| 45000 | 37.329 | 37.507 | 38.037 | 38.249 |
| 50000 | **37.458** | 37.622 | 38.102 | 38.290 |
| 55000 | 37.533 | 37.685 | 38.120 | 38.289 |
| 60000 | 37.626 | 37.767 | 38.168 | 38.321 |
| 65000 | 37.709 | 37.839 | 38.211 | 38.353 |
| 70000 | 37.768 | 37.894 | 38.235 | 38.366 |
| === PITQ mean queue length estimates === | | | | |
| Run Length | Run-In 0 | Run-In 500 | Run-In 2500 | Run-In 4500 |
| | | | | ** |
| 5000 | 2.712 | 2.889 | 3.204 | 3.324 |
| 10000 | 3.127 | 3.233 | 3.430 | **3.522** |
| 15000 | 3.284 | 3.359 | 3.497 | 3.559 |
| 20000 | 3.364 | 3.422 | **3.528** | 3.573 |
| 25000 | 3.411 | 3.457 | 3.544 | 3.579 |
| 30000 | 3.440 | 3.479 | 3.552 | 3.581 |
| 35000 | 3.464 | 3.498 | 3.560 | 3.585 |
| 40000 | 3.486 | **3.516** | 3.571 | 3.593 |
| 45000 | 3.495 | 3.521 | 3.570 | 3.589 |
| 50000 | 3.507 | 3.531 | 3.576 | 3.593 |
| 55000 | **3.515** | 3.537 | 3.577 | 3.592 |
| 60000 | 3.524 | 3.544 | 3.581 | 3.595 |
| 65000 | 3.532 | 3.551 | 3.585 | 3.598 |
| 70000 | 3.538 | 3.556 | 3.587 | 3.599 |

**Table F.15.** PITQ mean queuing time and mean queue length estimates as a function of the simulation run length and of the run-in-period.

| === LOADQ mean queuing time estimates === | | | | |
|---|---|---|---|---|
| Run Length | Run-In 0 | Run-In 500 | Run-In 2500 | Run-In 4500 |
| | | ** | | |
| 5000 | 14.754 | 12.081 | 11.602 | 11.435 |
| 10000 | 13.100 | 11.775 | 11.534 | 11.498 |
| 15000 | 12.567 | 11.686 | 11.529 | 11.508 |
| 20000 | 12.303 | 11.644 | 11.525 | 11.512 |
| 25000 | 12.136 | 11.609 | 11.515 | 11.504 |
| 30000 | 12.025 | 11.585 | 11.506 | 11.497 |
| 35000 | 11.953 | 11.577 | 11.509 | 11.502 |
| 40000 | 11.894 | 11.565 | 11.506 | 11.499 |
| 45000 | 11.849 | 11.557 | 11.504 | 11.498 |
| 50000 | 11.824 | 11.561 | 11.514 | 11.509 |
| 55000 | 11.799 | 11.560 | 11.517 | 11.513 |
| 60000 | 11.766 | 11.547 | 11.508 | 11.503 |
| 65000 | 11.743 | 11.541 | 11.505 | 11.501 |
| 70000 | 11.733 | 11.545 | 11.512 | 11.508 |

**Table F.16.** LOADQ mean queuing time estimates as a function of the simulation run length and of the run-in-period.

In Table F.16. we give only the mean queuing time estimates as the steady state mean queue length value is small and will not be used to test this procedure (See Section 2.6.1.). For the mean queuing time parameter we notice that using the run-in-period estimated in this appendix the parameter will reach the steady state for a simulated time of 10000 as compared to a simulation run length of 55000 that is required if no run-in-period is used. It can also be noticed that using a longer run-in-period, i.e. 4500, the parameter will reach the steady state for a shorter simulation run length (5000). But in this case, as in several other previous examples, the mean estimates obtained with the run-in-period estimated with our method are closer to the real steady state value $\mu$, than those obtained with a longer run-in-period. And at the same time we must keep in mind that we are not claiming that our method will give the **optimal**, but **good** results.

## F.4. ANALYSIS OF SOME SYSTEMS WITH KNOWN ANALYTICAL STEADY STATE VALUE.

In this appendix we analyse some results obtained for the M/M/4 queue and the Jackson's system discussed in Chapter 4. Both systems have known steady state values. Section F.4.1. will give the results for the M/M/4 queue, while section F.4.2. will study the Jackson's model results.

### F.4.1. M/M/4 QUEUE

#### a. Estimation of the run-in-period.

This system was modelled for an arrival rate $\lambda=1/15$ and a service rate $\mu=1/50$; for this case we define $\tau=\lambda/s\mu = (1/15)/(4*1/50)=0.833$ (s: number of servers, 4 in this example; $\tau<1$ for stable systems). From the equations given in Chapter 4 for the M/M/s queue, we can estimate $L_q$ and $W_q$:

$P(j\geq4) = ((s\tau)^s P(j \geq 0))/(1-\tau) = 0.6574;$

Therefore, $L_q = 0.6574x0.833/(0.1667)=3.287;$

Using Little's formula, $W_q = 3.2481/(1/15)=49.305.$

Figure F.9. shows a graph of the standard deviation of the mean queuing time estimate for the queue of customers waiting to be served as a function of the simulation run length for different number of replications. From this graph we notice that while for a small number of replications (20) there are two maximum values, when the number of replications is increased (200) there exists only one maximum that occurs at 2700. The problem of the influence of the number of replications is directly related to the large variance of the estimate. In practice this can be noticed from the estimates obtained from the simulation if besides obtaining estimates of the standard deviation we also estimate the mean corresponding to each one of the intervals of time, $j*T_1$. To give an example, Table F.17. gives the mean estimates and the standard deviation as a function of the simulation run length when the number of replications used is 20. From this table we notice that in the best case, the standard deviation is half the value of the mean. For a simulation run length of 15300 (data not shown in Table F.17.) the standard deviation is 0.66 of the mean estimate. In simulation models with a large standard deviation as compared to the mean estimate it will be necessary to use more than 20 or 30 replications, which may be a disadvantage of the method, although computer time is not the problem it was some years ago. Another way of identifying the problem of the influence of the number of replications is by studying the graph. When it has more than one local maximum, like in Figure F.9. when 20 replications or 40 replications are taken, the number of replications should be increased until only one maximum of the standard deviation is detected. A similar problem is illustrated in the following example for a two-stage queuing system. Therefore, we will use as an estimate of the run-in-period for this example of the M/M/4 queue a value of 2700.

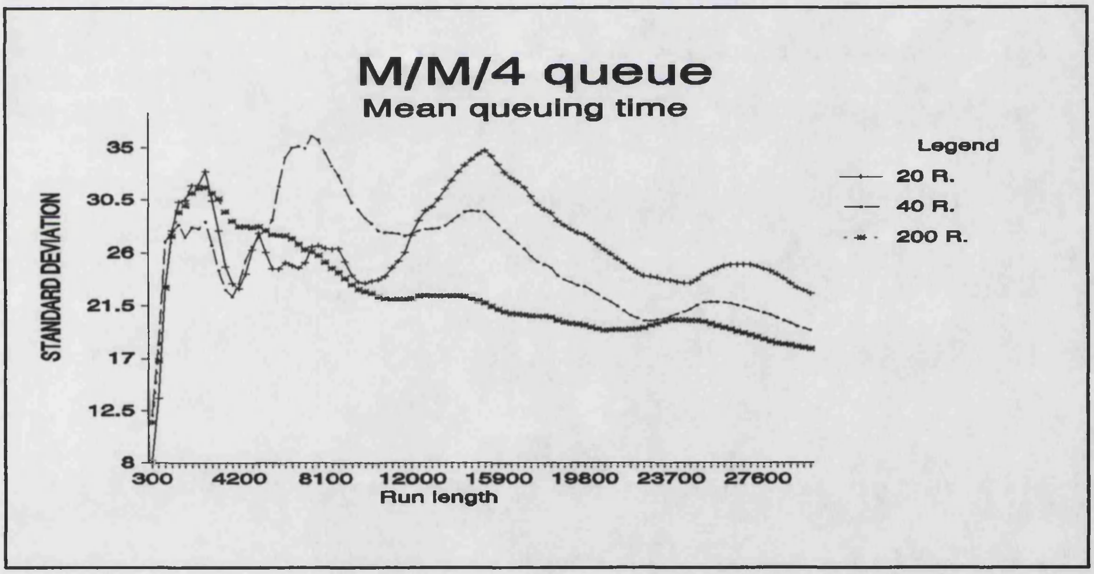



**Figure F.9.** STANDARD DEVIATION of the M/M/4 queue mean queuing time as a function of the simulation run length and of the number of replications.

| M/M/4 mean queuing time and standard deviation estimates | | | | | |
|---|---|---|---|---|---|
| Run length | Mean | Std.Dev. | Run length | Mean | Std.Dev. |
| 150 | 7.586 | 8.237 | 3150 | 42.69 | 25.18 |
| 300 | 17.270 | 13.575 | 3300 | 42.87 | 24.91 |
| 450 | 24.696 | 23.229 | 3450 | 43.06 | 24.72 |
| 600 | 28.941 | 27.887 | 3600 | 43.62 | 25.19 |
| 750 | 30.665 | 30.303 | 3750 | 44.36 | 26.54 |
| 900 | 32.822 | 30.483 | 3900 | 44.50 | 26.66 |
| 1050 | 35.101 | 31.749 | 4050 | 44.34 | 26.39 |
| 1200 | 36.822 | 31.730 | 4200 | 44.55 | 26.29 |
| 1350 | 38.003 | 32.933 | 4350 | 44.73 | 26.34 |
| 1500 | 37.455 | 30.447 | 4500 | 44.79 | 25.26 |
| 1650 | 36.722 | 27.837 | 4650 | 44.98 | 24.06 |
| 1800 | 36.880 | 24.829 | 4800 | 45.26 | 23.48 |
| 1950 | 38.053 | 23.356 | 4950 | 45.89 | 23.38 |
| 2100 | 39.210 | 22.959 | 5100 | 46.17 | 23.57 |
| 2250 | 39.405 | 24.224 | 5250 | 46.29 | 23.72 |
| 2400 | 40.478 | 26.270 | 5400 | 46.59 | 24.00 |
| 2550 | 40.230 | 27.627 | 5550 | 47.21 | 24.74 |
| 2700 | 39.766 | 26.058 | 5700 | 47.64 | 25.30 |
| 2850 | 39.959 | 24.681 | 5850 | 48.36 | 26.04 |
| 3000 | 41.193 | 24.648 | 6000 | 48.79 | 27.50 |

**Table F.17.** M/M/4 mean queuing time and corresponding standard deviation mean estimates as a function of the simulation run length.

### b. Evaluation of the run-in-period performance

Tables F.18a. and F.18b give the mean queuing time and the mean queue length estimates for the M/M/4 queue, as a function of the simulation run length, and as a function of the run-in-period.

Using a value of $\epsilon=2.5\%$ the parameters will be considered to be in the steady state if the estimates fall in the following ranges:

| **Parameter** | **Range** |
|---|---|
| Queuing time | [48.072 , 50.537] |
| Queue length | [3.205 , 3.369] |

| M/M/4 mean queuing time estimates | | | | |
|---|---|---|---|---|
| Run Length | Run-In 0 | Run-In 1300 | Run-In 2700 | Run-In 7650 |
| 1500 | 29.824 | 39.577 | | |
| 2000 | 33.507 | 42.221 | | |
| 2500 | 35.965 | 43.517 | | |
| 3000 | 37.738 | 44.302 | 43.356 | |
| 3500 | 38.961 | 44.701 | 44.730 | |
| 4000 | 39.929 | 45.075 | 45.385 | |
| 4500 | 40.901 | 45.625 | 46.162 | |
| 5000 | 41.601 | 45.905 | 46.739 | |
| 5500 | 42.197 | 46.178 | 47.244 | |
| 6000 | 42.609 | 46.274 | 47.380 | |
| 6500 | 43.198 | 46.658 | 47.931 | |
| 7000 | 43.547 | 46.788 | 48.152 | |
| 7500 | 43.873 | 46.922 | 48.250 | |
| 8000 | 44.155 | 47.025 | 48.227 | 45.493 |
| 8500 | 44.419 | 47.139 | 48.208 | 46.018 |
| 9000 | 44.606 | 47.179 | 48.146 | 46.464 |
| 9500 | 44.833 | 47.284 | 48.163 | 47.009 |
| 10000 | 45.139 | 47.494 | 48.315 | 47.663 |
| 10500 | 45.376 | 47.641 | 48.443 | 48.036 |
| 11000 | 45.633 | 47.818 | 48.606 | 48.452 |
| 11500 | 45.903 | 48.015 | 48.789 | 48.837 |
| 12000 | 46.046 | 48.076 | 48.830 | 48.893 |
| 12500 | 46.184 | 48.137 | 48.830 | 48.923 |
| 13000 | 46.362 | 48.250 | 48.910 | 49.081 |
| 13500 | 46.514 | 48.338 | 48.971 | 49.168 |
| 14000 | 46.641 | 48.406 | 48.988 | 49.241 |

**Table F.18a.** Mean queuing time estimates for the M/M/4 queue as a function of the simulation run length and for different run-in-periods.

| M/M/4 mean queue length estimates | | | | |
|---|---|---|---|---|
| Run Length | Run-In 0 | Run-In 1300 | Run-In 2700 | Run-In 7650 |
| 1500 | 2.118 | 2.904 | | |
| 2000 | 2.349 | 3.001 | | |
| 2500 | 2.499 | 3.043 | | |
| 3000 | 2.607 | 3.072 | 3.120 | |
| 3500 | 2.684 | 3.089 | 3.173 | |
| 4000 | 2.739 | 3.096 | 3.172 | |
| 4500 | 2.799 | 3.125 | 3.214 | |
| 5000 | 2.843 | 3.139 | 3.238 | |
| 5500 | 2.876 | 3.147 | 3.252 | |
| 6000 | 2.896 | 3.144 | 3.245 | |
| 6500 | 2.930 | 3.163 | 3.271 | |
| 7000 | 2.950 | 3.167 | 3.274 | |
| 7500 | 2.967 | 3.170 | 3.272 | |
| 8000 | 2.985 | 3.176 | 3.269 | 3.267 |
| 8500 | 2.999 | 3.179 | 3.260 | 3.239 |
| 9000 | 3.011 | 3.182 | 3.253 | 3.232 |
| 9500 | 3.025 | 3.188 | 3.253 | 3.247 |
| 10000 | 3.044 | 3.200 | 3.261 | 3.280 |
| 10500 | 3.058 | 3.208 | 3.267 | 3.291 |
| 11000 | 3.074 | 3.218 | 3.276 | 3.308 |
| 11500 | 3.090 | 3.229 | 3.285 | 3.325 |
| 12000 | 3.099 | 3.233 | 3.287 | 3.324 |
| 12500 | 3.107 | 3.235 | 3.286 | 3.320 |
| 13000 | 3.117 | 3.242 | 3.289 | 3.326 |
| 13500 | 3.126 | 3.246 | 3.291 | 3.328 |
| 14000 | 3.134 | 3.250 | 3.292 | 3.330 |

**Table F.18b.** Mean queue length estimates for the M/M/4 queue as a function of the simulation run length and for different run-in-periods.

The run-in-period using Gordon's method has been estimated as 7650 (See Appendix E). From the underlined values in the tables we notice that both the mean queuing time and the mean queue length estimates will reach the steady state for shorter simulation run lengths if the run-in-period estimated with our method is used than if the run-in-period estimated with Gordon's method is used. As in most previous examples the run-in-period estimated with Gordon's method is overestimated. On the other hand, if

Conway's method is used the estimated run-in-period is 300 which is clearly underestimated.

### F.4.2. 2-STAGE QUEUING SYSTEM

This example given in Appendix A is repeated here for convenience:

"The last two things that are done to a car before its manufacture is complete are installing the engine and putting on the tires. An average of 60 cars per hour arrive, and there is only one worker for the installation of the engine; he can serve an average of 54 cars per hour. There are three available workers for putting on the tyres, and each one can serve 162 cars per hour." (Winston, 1987).

For stage 1, the number of servers, $s_1 = 1$; the service rate $\mu_1 = 1/54$; and the arrival rate $\lambda = 1/60$; for stage 2, $s_2 = 3$; $\mu_2 = 1/162$;

The system is stable as $\tau_1 = \lambda/\mu_1 = 54/60 = 0.90 < 1$; Similarly, $\tau_2 = \lambda/(3^*\mu_2) = 162/180 < 1$; therefore, the system has a steady state solution.

For stage 1, $L_q = \tau^2/(1-\tau) = 8.1$ cars waiting for the engine to be installed. Using Little's equation, $W_q = 8.1/(1/60) = 486$.

For stage 2, $P(j \geq 3)=0.83$. (j:number of cars in stage 2); $\tau_2 = 0.90$; therefore, $L_q = 0.83^*0.90/(1-0.90) = 7.47$ cars. Using Little's formula, $W_q=420$.

#### a. Estimation and evaluation of the performance of the run-in-period

For this system, only the queue of the cars waiting for the ENGINE to be installed has been studied using simulation (See Figure A.11, Appendix A). In this case the traffic intensity is 0.9. But from some empirical results we have found that queues with traffic intensity values of 0.9 or greater are not well behaved. Although this requires further research, some previous results show that the approach to the steady state is cyclic and not completely monotonic as in other queues with smaller values of traffic intensity. And from this example as shown in Figure F.10. the run length for which the standard deviation of the mean queuing time estimates reaches its maximum is greatly influenced by the number of replications.

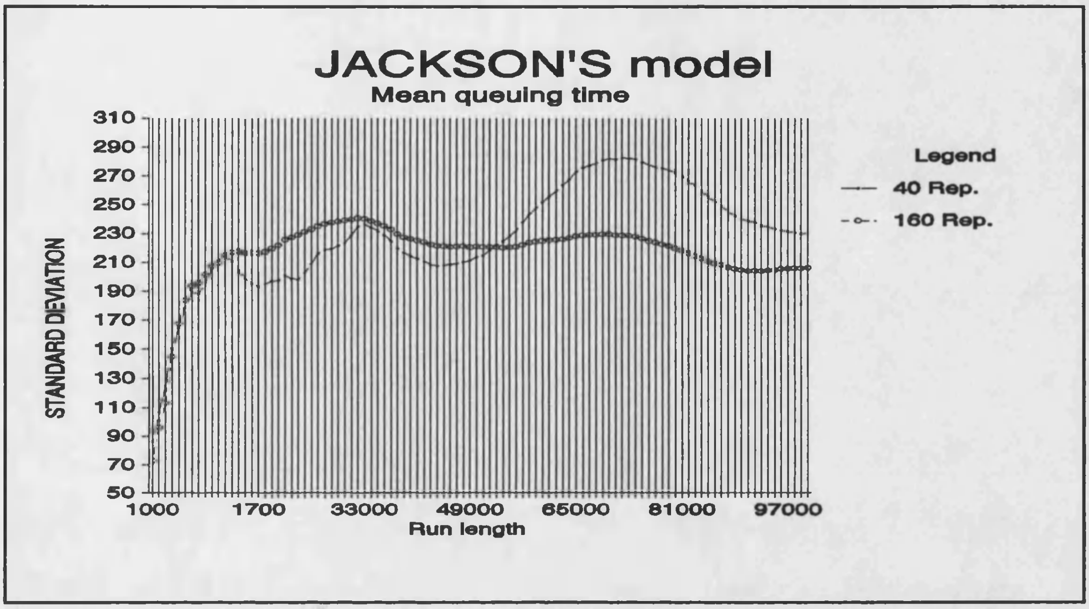



**Figure F.10.** STANDARD DEVIATION of the ENGINE mean queuing time estimates as a function of the simulation run length and of the number of replications.

When 40 replications are used several local maximum exist, and the one with the largest value corresponds to a simulation run length of 72500. When a very large number of replications (160) is used, the standard deviation graph shows only one maximum that corresponds to a simulation run length of 33000. A similar graph can be drawn for the mean queue length parameter. As discussed in Section F.4.3. for the M/M/4 queue, when for a small number of replications the graph of the standard deviation shows several local maximum points, it is necessary to increase the number of replications, or the run-in-period that we estimate may be overestimated.

Mean values have been obtained for different simulation run lengths and different run-in-periods, and they are given in Tables F.19a. and F.19b.

| JACKSON'S model : Mean queuing time estimates | | | | |
|---|---|---|---|---|
| Run Length | Run-In 0 | Run-In 12000 | Run-In 33000 | Run-In 72500 |
| 5000 | 217.727 | | | |
| 10000 | 288.754 | | | |
| 15000 | 329.325 | 404.855 | | |
| 20000 | 356.268 | 421.899 | | |
| 25000 | 376.750 | 434.605 | | |
| 30000 | 392.371 | 444.119 | | |
| 35000 | 402.754 | 448.359 | | |
| 40000 | 411.580 | 452.549 | 460.302 | |
| 45000 | 419.991 | 457.808 | 471.543 | |
| 50000 | 426.749 | 461.579 | 475.864 | |
| 55000 | 431.832 | 463.921 | 476.801 | |
| 60000 | 435.950 | 465.668 | 477.200 | |
| 65000 | 440.236 | 468.110 | 479.289 | |
| 70000 | 444.396 | 470.717 | 482.058 | |
| 75000 | 448.146 | 473.048 | 484.379 | 483.616 |
| 80000 | 451.179 | 474.764 | 485.611 | 485.069 |
| 85000 | 453.508 | 475.848 | 486.064 | 484.708 |
| 90000 | 455.855 | 477.116 | 486.991 | 487.065 |
| 95000 | 458.082 | 478.375 | 487.937 | 488.779 |
| 100000 | 460.053 | 479.468 | 488.684 | 489.836 |
| 105000 | 462.525 | 481.213 | 490.385 | 493.057 |
| 110000 | 464.993 | 483.031 | 492.134 | 495.909 |

**Table F.19a.** Mean queuing time estimates for the JACKSON MODEL as a function of the simulation run length and of different run-in-periods.

| JACKSON'S model : Mean queue length estimates | | | | |
|---|---|---|---|---|
| Run Length | Run-In 0 | Run-In 12000 | Run-In 33000 | Run-In 72500 |
| 5000 | 3.778 | | | |
| 10000 | 4.938 | | | |
| 15000 | 5.607 | 7.060 | | |
| 20000 | 6.051 | 7.262 | | |
| 25000 | 6.380 | 7.429 | | |
| 30000 | 6.621 | 7.539 | | |
| 35000 | 6.782 | 7.584 | | |
| 40000 | 6.928 | 7.650 | 8.000 | |
| 45000 | 7.064 | 7.726 | 8.091 | |
| 50000 | 7.171 | 7.779 | 8.105 | |
| 55000 | 7.251 | 7.810 | 8.090 | |
| 60000 | 7.315 | 7.833 | 8.076 | |
| 65000 | 7.387 | 7.871 | 8.105 | |
| 70000 | 7.454 | 7.911 | 8.139 | |
| 75000 | 7.515 | 7.948 | 8.169 | 8.408 |
| 80000 | 7.565 | 7.974 | 8.185 | 8.334 |
| 85000 | 7.600 | 7.987 | 8.184 | 8.262 |
| 90000 | 7.643 | 8.012 | 8.200 | 8.292 |
| 95000 | 7.680 | 8.032 | 8.213 | 8.307 |
| 100000 | 7.713 | 8.049 | 8.223 | 8.312 |
| 105000 | 7.751 | 8.074 | 8.244 | 8.345 |
| 110000 | 7.790 | 8.101 | 8.269 | 8.380 |
| 115000 | 7.803 | 8.101 | 8.257 | 8.345 |

**Table F.19b.** Mean queue length estimates for the JACKSON MODEL as a function of the simulation run length and of different run-in-periods.

We will consider the parameters to have reached the steady state if the mean estimates fall in the following ranges:

| Parameter | Range |
|---|---|
| Queuing time | [473.80 , 498.10] |
| Queue length | [7.90 , 8.30] |

In Tables F.19a. and F.19b we have underlined the mean estimates beyond which the mean estimates all fall in these ranges. As can be seen from this table a run-in-period of 33000 gives mean queuing time estimates that are closer to the real value of 486, than those obtained with shorter run-in-periods. A run-in-period of 72500 estimated when the number of replications is 40 is clearly overestimated.

In Appendix E we estimated a run-in-period of approximately 114700 for the mean queuing time and the mean queue length of the queue of CARS waiting for the ENGINE to be installed when Gordon's method is used. As in most cases before, this run-in-period is overestimated, as can be seen from the results in Tables F.19a. and F.19b.

## F.5. CONCLUSIONS

As has been seen from the examples included in this appendix, the run-in-period estimated with the method proposed in Section 4.4.3. usually gives mean estimates that reach the steady state for shorter simulation run lengths than other run-in-periods, shorter or longer. When the mean estimates obtained using a shorter run-in-period reach the steady state for the same simulation run length then those estimates obtained with the run-in-period estimated with our method will be closer to the real steady state value, $\mu$. In some cases, there are run-in-periods for which the mean estimates will reach the steady state for a shorter simulation run length than when the estimated run-in-period is used. In these cases, the difference is small, and this does not invalidate our procedure. There is always some uncertainty associated with the output of the simulation, and for this reason we do not claim that our method will always give the best results, but as we showed in the examples of both this appendix and of Chapter 4, the steady state is reached sooner if the run-in-period estimated with our method is used than if no run-in-period is used. In some cases, the difference will be significative, while in others, the use of a run-in-period will not shorten the simulation run length required for the parameter to reach the steady state in a significative way. This is part of the analysis previous to the simulation, as was discussed in Chapter 2 (section 2.5.2.).

There may be a problem with the method when the system, or the parameter of interest has a large variance as compared to the mean values. In this case, it may be necessary to make more than 20 or 30 replications. The problem will be easily identified because the graph of the standard deviation when the number of replications is small will show several local maximum points. Selecting the largest of these local maximum points will give an overestimated run-in-period.

# APPENDIX G : ADDITIONAL RESULTS FOR CHAPTER 5.

## G.1. INTRODUCTION

Chapter 5 discusses the **BATCH MEANS METHOD.** In Section G.2. we summarise the points discussed in section 5.5.2. concerning the application of the method to complex simulation models. Additional results concerning the discussion of Chapter 5 are given in Section G.3.

## G.2. STEADY STATE SIMULATION

Section G.2.1. gives a summary of the discussion of this method presented in Section 5.5. In section G.2.2. we explain the terminology used in this research, as its meaning is slightly different from that found in the literature.

### G.2.1. DISCUSSION OF THE BATCH MEANS METHOD.

From the empirical results shown in Chapter 5 and in this Appendix we show the following:

1. The number of batches is not critical as suggested by some authors (see Law and Carson (1979) and Schmeiser, (1982)). What is important is the total number of observations, or total simulation run length, and the lack of correlation of the batch means.

2. There is a value $m^*$ for the number of observations required if the estimate is going to be accurate. This value of $m^*$ is in general unknown but its approximate value can be easily determined using a graphical method. A number of observations $m<m^*$ will give a biased estimator.

3. Descriptive Sampling (D.S.) can be used to estimate steady state parameters using the batch means method. Due to its smaller standard deviation, the total number of observations that are required to obtain a non-

biased mean estimate is less than the number required when Random Sampling (R.S) is used.

### G.2.2. MODIFICATION OF THE BATCH MEANS METHOD

In the experiments reported in this thesis the batch means method has been slightly modified as will be explained. Instead of recording N observations and grouping them into B batches, a simulation run length $T_0$ is chosen. This simulation run length should be sufficiently long for the system to reach the steady state. We divide this run length into B equal intervals with a sub-run length of $T_1 = T_0/B$. The batch means $\bar{X}j$ (j=1, 2,..B) are obtained as the average of the observations recorded during the interval of time $[(j-1)*T_1, j*T_1]$. The estimate $\bar{Y}$ is given by the average of these batch means $\bar{X}j$:

$$\bar{Y} = \sum_{j=1}^{B} \frac{\bar{X}_j}{B} \qquad \text{(G.1)}$$

Therefore, with this modification, the batch size, defined before as the number of observations in each batch, is a random number. However, the method still works well. To use the same terminology used with the batch means method, the term "batch size" will refer to the sub-run length $T_1$.

This means that with this modification, instead of trying to determine the value of m* (See Section G.2.1.) we will try to determine a value R* for the minimum simulation run length for which the estimate converge to the steady state value.

## G.3. ANALYSIS OF THE RESULTS

In this section we will describe the experiments carried out to illustrate the different points of section G.2.1. and we will analyse the results obtained for some queues the PUB and the STEELWORKS models.

### G.3.1. DESCRIPTION OF THE EXPERIMENTS

In the experiments carried out to confirm the points discussed in Section G.2.2. we chose a long simulation run length and divided it into different subrun lengths; the value of these subrun lengths is such that 10, 20, 30, 50 and 100 batch means are obtained for each one of the parameters of interest (queuing time or queue length of some of the queues of the model).

The estimate $\overline{Y}$ is calculated from equation G.1., where $\overline{X}_i$ is the value of the mean in batch i, and B is the number of batches (10, 20, 30, 50 or 100).

For each estimate, the standard deviation, and the confidence interval relative precision (at a 95% confidence level) were also calculated.

As pointed out in Section G.2.2. there exists a minimum simulation run length R* for the mean estimate to show clearly the convergence to the steady state value. To check this point in an empirical way we obtained not only the batch mean corresponding to B batches, but also the batch mean corresponding to B1 batches, where B1 takes values 2, 3 ... B. The batch mean in this case is easily calculated using equation (G.2).

$$\overline{Y}(B1,B)=\frac{\sum_{i=1}^{B1}\overline{X}_i}{B1} \qquad \text{(G.2)}$$

Calling $T_0$ the total simulation run length (for example 15'000000 for the PUB model), $\overline{Y}$(B1,B) can be interpreted as the batch mean for B1 batches where each batch has size $T_0$/B. Studying these values it is possible to determine the approximate number of batches for which the mean estimates converge to the steady state value.

### G.3.2. ANALYSIS OF THE RESULTS.

In this sub-section we analyse results obtained for the PUB and the STEELWORKS models.

## 1. The PUB

Simulation output data was recorded for the following queues:

1. WAIT.
2. CLEAN.
3. IDLE.

As in the case of the LAUNDERETTE both the queuing time and the queue length batch means were estimated. A simulation run length of 15'000000 was chosen; this run length was divided into sub-runs lengths of 150000, 300000, 500000, 750000 and 1'500000. In this way we obtain 100, 50, 30, 20 and 10 batches. For each batch the mean was estimated and the average of all the batch means gives the grand mean $\bar{Y}$. As in the study of the LAUNDERETTE model carried out in Chapter 5, we will illustrate with empirical results the four points discussed in section G.2.2. These results, similar to those obtained for the LAUNDERETTE, corroborate the conclusions given in Chapter 5 concerning the use of the **batch means** method.

We now discuss each one of these points.

**1. Minimum simulation run length required for the parameter that is being estimated to reach the steady state.**

To show the point that there exists for each different parameter, i.e., queuing time or queue length, a minimum simulation run length required for the parameter to reach the steady state, we obtained the mean estimate $\bar{Y}$, as a function of the simulation run length, using independent random number seeds, and a batch size of 300000. Figure G.1. shows the batch mean estimates for the mean queuing time parameter of the WAIT queue as a function of the simulation run length. It can be seen from this graph that the mean estimates converge to the mean esteady state value of 1.141 (obtained in Appendix C) but that for this convergence a simulation run length of at least 18'000000 is required.

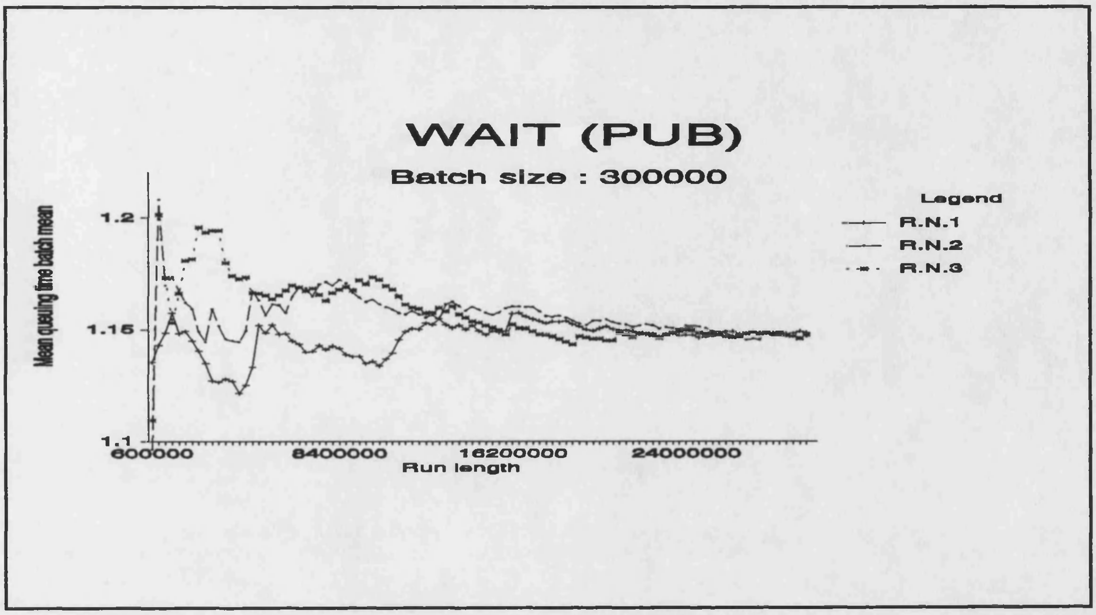



**Figure G.1.** WAIT batch mean queuing time estimates as a function of the simulation run length and of the random number seeds.

However, as explained in Chapter 5, in most practical simulations the practitioner does not require to estimate the parameter with this accuracy and a certain tolerance, for example of 2.5% is allowed. In this case a good stopping criterion is the value of the c.i. relative precision. Figure G.2. shows the c.i. limits for different simulation run lengths. It can be seen how, as expected, they cover the real steady state value and how, as expected also, the width of the 95% c.i. decreases as the simulation run length increases. Although the results are not shown here, we obtained 100 independent batch mean estimates and the coverage of the c.i. is close to the desired value of 1-$\alpha$, where we chose a value $\alpha$ = 0.05.

Therefore, the procedure to follow when the batch means method is used for the estimation of steady state parameter is to obtain a batch at a time and to check if the c.i. relative precision is smaller than the desired precision. The simulation run is stopped when this condition is met. Obviously, a test should be performed to test that the batch size is such that the batch means are uncorrelated.

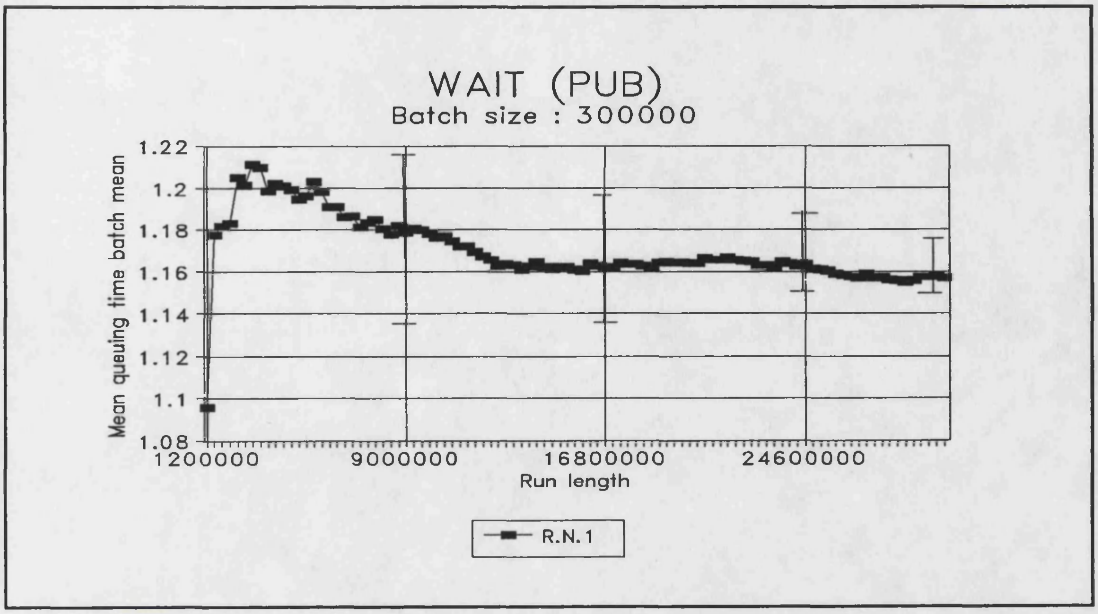



**Figure G.2.** WAIT batch mean queuing time estimates as a function of the simulation run length and c.i. limits for different simulation run lengths.

**2. D.S. BATCH MEAN estimates will reach the steady state for shorter simulation run lengths.**

In order to test the performance of D.S. when applied to the batch means method, we used different batch sizes and obtained the mean queuing time estimates for the queues of interest in this model. The results are shown in Figures G.3.(WAIT), G.4. (CLEAN) and G.5. (IDLE), for a batch size of 500000; these graphs also show the results obtained for R.S. Results for other batch sizes provide the same information and are not given in this appendix.

It can be seen from these figures that the D.S. batch mean estimates converge to the steady state for shorter simulation run lengths; although results for other batch sizes are not given this conclusion is valid independent of the batch size.

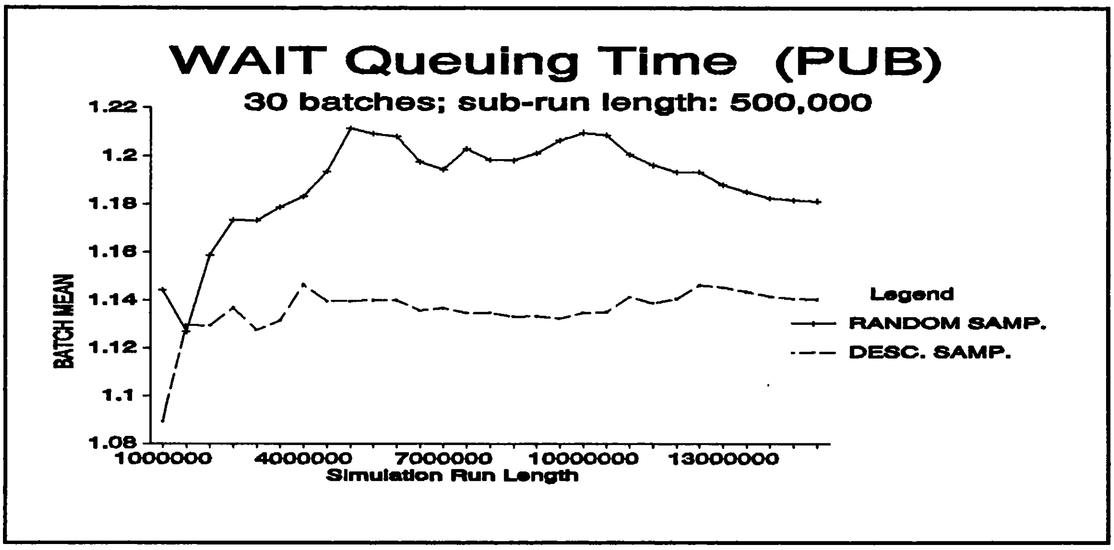



**Figure G.3.** WAIT mean queuing time batch means estimates as a function of the simulation run length and of the sampling method. The total simulation run length has been divided into 30 batches.

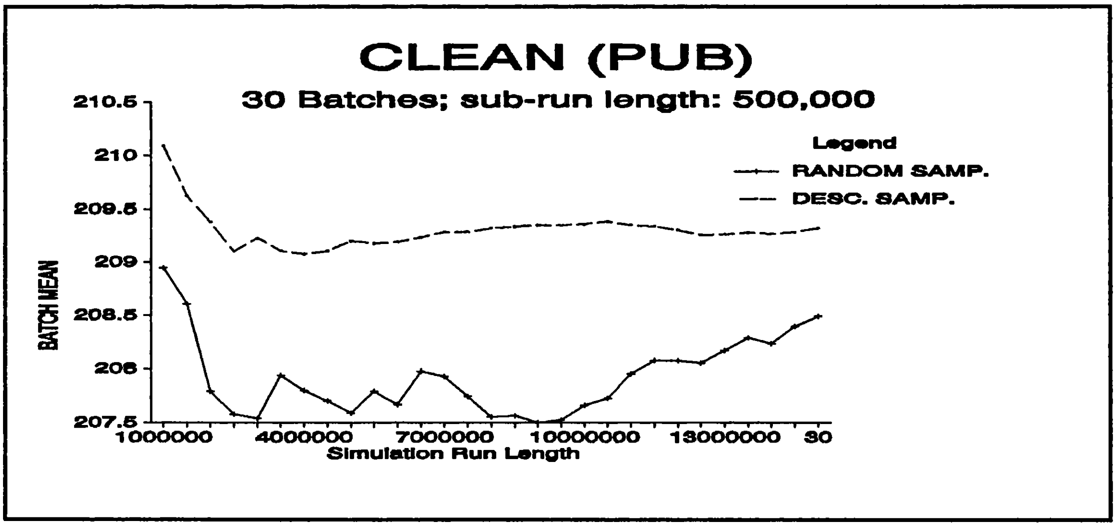



**Figure G.4.** CLEAN queuing time batch means estimates as a function of the simulation run length and of the sampling method, when the total simulation run length has been divided in 30 batches.

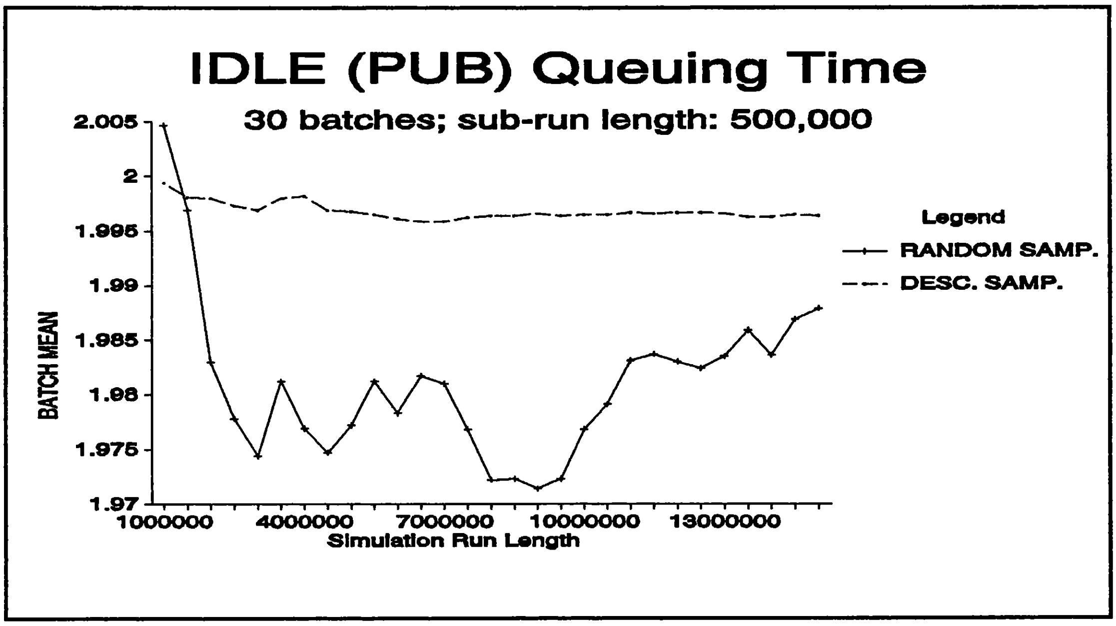



Figure G.5. IDLE queuing time batch means estimates as a function of the simulation run length and of the sampling method, when the total simulation run length has been divided in 30 batches.

This implies that if D.S. is used, we obtain, with shorter simulation run lengths and with a smaller number of batches, estimates as accurate and close to the real value, $\mu$, as those obtained with R.S. for longer simulation run lengths and greater number of batches.

Although the values of the standard deviation corresponding to the batch means given as a function of the simulation run length are not shown, if we compare the R.S. and the D.S. values we would expect the standard deviation for the D.S estimates to be smaller and therefore the c.i. half width will be shorter; as the point estimate is approximately the same, the D.S. c.i. relative precision will be smaller and therefore, the simulation run length required to obtain an estimate with a given precision will be shorter if D.S. is used than if R.S. is used.

### 3. Influence of the number of batches.

Table G.1. summarises the results obtained for different batch sizes and the different queues. It gives the batch mean based on 10, 30 and 100 batches for a total simulation run length of 15'000,000, and the corresponding standard deviation and the c.i. relative precision for both Random and Descriptive Sampling. The table also gives the mean estimates obtained with the method of replications when the run length is 120000.

We can compare the values in table G.1. obtained for a total simulation run length of 15'000000 to the real steady state values (obtained in Appendix C); we notice that, as in the case of the LAUNDERETTE model, the difference between the Descriptive Sampling mean estimates and the corresponding value of $\mu$ is negligible in most cases.

From table G.1. we may conclude that for a total simulation run length of 15'000000 it is valid that:

1. The point estimate (mean queuing time or mean queue length) is approximately the same, independent of the number of batches.

2. The relative precision of the confidence interval is approximately the same for a given parameter, independent of the number of batches.

3. When the simulation run length is very long, and it needs to be very long for the point estimate not to be biased, the c.i. half-width and the c.i. relative precision are smaller than when the replications method is used.

4. Comparing the batch means results with the steady state values ($\mu$) obtained from the results of Appendix C, the D.S. batch means estimates are closer to the real steady state value than the Random Sampling estimates. In most cases the percentage error of the Descriptive Sampling batch mean estimates is almost zero.

| RANDOM SAMPLING | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 10 Batches | | | 30 Batches | | | 100 Batches | | | |
| Queue | Parameter | Batch | Std. Dev. | C.I. Half | Batch | Std. Dev. | C.I. Half | Batch | Std. Dev. | C.I. Half | |
| WAIT | Queue. | 1.181 | 0.059 | 0.036 | 1.181 | 0.081 | 0.026 | 1.180 | 0.158 | 0.027 | |
| WAIT | Queue | 0.237 | 0.012 | 0.036 | 0.237 | 0.017 | 0.026 | 0.237 | 0.033 | 0.028 | |
| CLEAN | Queue. | 208.482 | 1.495 | 0.005 | 208.490 | 2.330 | 0.004 | 208.533 | 4.751 | 0.005 | |
| CLEAN | Queue | 41.766 | 0.185 | 0.003 | 41.766 | 0.260 | 0.002 | 41.766 | 0.500 | 0.002 | |
| IDLE | Queue. | 1.988 | 0.027 | 0.010 | 1.988 | 0.045 | 0.009 | 1.989 | 0.093 | 0.009 | |
| IDLE | Queue | 0.796 | 0.008 | 0.007 | 0.796 | 0.013 | 0.006 | 0.796 | 0.027 | 0.007 | |
| | | | | | | | | | | | |
| | | 10 Batches | | | 30 Batches | | | 100 Batches | | | |
| Queue | Parameter | Batch | Std. Dev. | C.I. Half | Batch | Std. Dev. | C.I. Half | Batch | Std. Dev. | C.I. Half | |
| WAIT | Queue. | 1.134 | 0.018 | 0.011 | 1.140 | 0.054 | 0.018 | 1.138 | 0.103 | 0.018 | |
| WAIT | Queue | 0.227 | 0.004 | 0.011 | 0.228 | 0.011 | 0.018 | 0.228 | 0.021 | 0.018 | |
| CLEAN | Queue. | 209.368 | 0.370 | 0.001 | 209.320 | 0.635 | 0.001 | 209.414 | 1.536 | 0.002 | |
| CLEAN | Queue | 41.880 | 0.074 | 0.001 | 41.884 | 0.133 | 0.001 | 41.897 | 0.296 | 0.001 | |
| IDLE | Queue. | 1.999 | 0.002 | 0.001 | 1.996 | 0.003 | 0.001 | 1.998 | 0.011 | 0.001 | |
| IDLE | Queue | 0.800 | 0.001 | 0.000 | 0.799 | 0.001 | 0.001 | 0.799 | 0.003 | 0.001 | |
| | | | | | | | | | | | |
| | | 30 Batches | | | | | | | | | |
| Queue | Parameter | Batch | Std. Dev. | C.I. Half | | | | | | | |
| WAIT | Queue. | 1.162 | 0.052 | 0.017 | | | | | | | |
| WAIT | Queue | 0.233 | 0.011 | 0.017 | | | | | | | |
| CLEAN | Queue. | 208.771 | 1.508 | 0.003 | | | | | | | |
| CLEAN | Queue | 41.804 | 0.158 | 0.001 | | | | | | | |
| IDLE | Queue. | 1.991 | 0.031 | 0.006 | | | | | | | |
| IDLE | Queue | 0.797 | 0.009 | 0.004 | | | | | | | |
| | | | | | | | | | | | |
| | | Mean | Std. Dev. | C.I. Half | | | | | | | |
| WAIT | Queue. | 1.152 | 0.197 | 0.019 | | | | | | | |
| WAIT | Queue | 0.230 | 0.038 | 0.019 | | | | | | | |
| CLEAN | Queue. | 209.250 | 5.649 | 0.003 | | | | | | | |
| CLEAN | Queue | 41.856 | 0.589 | 0.002 | | | | | | | |
| IDLE | Queue. | 2.000 | 0.105 | 0.006 | | | | | | | |
| IDLE | Queue | 0.800 | 0.030 | 0.004 | | | | | | | |

**Table G.1.** Batch mean and replications mean queuing time and mean queue length estimates when the total simulation run length for the batch mean estimates is 15'000000. Results are given for the total simulation run length divided into 10, 30 and 100 batches, and also for a total simulation run length of 30'000000 divided into 30 batches.

## 2. THE STEELWORKS

Three queues of these model have been analysed:

1. TBLOWQ.
2. PITQ.
3. LOADQ.

Similar to Table G.1. the results comparing the batch means using R.S. and D.S., and the sample mean based on 300 replications, are given in Table G.2. The analysis of these results is similar to those already done, and the conclusions obtained from Table G.1. are valid also for the results of Table G.2.

| RANDOM SAMPLING | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Queue | Paramete | 10 Batches | | | 30 Batches | | | 100 Batches | | |
| | | Batch | Std. Dev | C.I. Half | Batch | Std. Dev | C.I. Half | Batch | Std. Dev | C.I. Half |
| TBLOW | QueueTi | 81.2822 | 1.29 | 0.0114 | 81.1607 | 2.7966 | 0.0129 | 81.2603 | 3.7951 | 0.0093 |
| TBLOW | Queue | 2.6642 | 0.0436 | 0.0117 | 2.6604 | 0.0938 | 0.0132 | 2.6636 | 0.127 | 0.0095 |
| PITQ | Queue. | 38.5018 | 0.4764 | 0.0089 | 38.5363 | 1.0025 | 0.0097 | 38.5018 | 1.3483 | 0.007 |
| PITQ | Queue | 3.6117 | 0.0445 | 0.0088 | 3.6153 | 0.0945 | 0.0098 | 3.6122 | 0.1279 | 0.007 |
| LOAD | Queue. | 11.4923 | 0.0386 | 0.0024 | 11.4882 | 0.0837 | 0.0027 | 11.49255 | 0.127 | 0.0022 |
| LOAD | Queue | 0.7052 | 0.0021 | 0.0021 | 0.705 | 0.0047 | 0.0025 | 0.7052 | 0.007 | 0.002 |
| | | | | | | | | | | |
| Queue | Paramete | 10 Batches | | | 30 Batches | | | 100 Batches | | |
| | | Batch | Std. Dev | C.I. Half | Batch | Std. Dev | C.I. Half | Batch | Std. Dev | C.I. Half |
| TBLOW | Queue. | 81.2613 | 0.6285 | 0.0055 | 81.4282 | 1.0426 | 0.0048 | 81.2603 | 3.7951 | 0.0093 |
| TBLOW | Queue | 2.6633 | 0.0203 | 0.0055 | 2.6696 | 0.0342 | 0.0048 | 2.6636 | 0.127 | 0.0095 |
| PITQ | Queue. | 38.5192 | 0.2143 | 0.004 | 38.4367 | 0.3679 | 0.0036 | 38.5018 | 1.3483 | 0.007 |
| PITQ | Queue | 3.6125 | 0.0203 | 0.004 | 3.6058 | 0.0344 | 0.0036 | 3.6122 | 0.1279 | 0.007 |
| LOAD | Queue. | 11.512 | 0.0191 | 0.0012 | 11.4972 | 0.025 | 0.0008 | 11.49255 | 0.127 | 0.0022 |
| LOAD | Queue | 0.7062 | 0.001 | 0.001 | 0.7054 | 0.0013 | 0.0007 | 0.7052 | 0.007 | 0.002 |
| REPLICATIONS METHOD (Run Length: 150000) | | | | | | | | | | |
| | | Mean | Std. Dev | C.I. Half | | | | | | |
| TBLOW | Queue. | 82.0947 | 5.3465 | 0.0074 | | | | | | |
| TBLOW | Queue | 2.6911 | 0.178 | 0.0075 | | | | | | |
| PITQ | Queue. | 38.2518 | 1.877 | 0.0056 | | | | | | |
| PITQ | Queue | 3.5858 | 0.1783 | 0.0056 | | | | | | |
| LOAD | Queue. | 11.6082 | 0.1653 | 0.0016 | | | | | | |
| LOAD | Queue | 0.7114 | 0.009 | 0.0014 | | | | | | |

**Table G.2.** Batch mean and replications mean queuing time an dmean queue length estimates when the total simulation run length for the batch means estimates is 50'000000. Results are given for the total simulation run length divided into 10, 30 and 100 batches. The total simulation run length for the method of replications is 120000.

## APPENDIX H : TESTS FOR CORRELATION OF SIMULATION OUTPUT DATA

### H.1. INTRODUCTION

As explained in Section 5.5.1. if the batch size is not large enough, the data may show autocorrelation. Therefore it is important after choosing a batch size (or subrun length) to test for independence of the subrun responses. Several tests have been proposed in the literature. Law and Kelton (1982) suggest a method based on the estimated lag 1 autocorrelation between at least 400 batch means. If this lag 1 autocorrelation is smaller than a constant c (suggested value: c=0.4.) the batch means can be considered independent. Fishman suggests a tests based on the Von Neumann statistic. However, empirical tests performed on this method seem to show that it has a poor performance. Kleijnen (1982) discusses the possible reasons for this poor performance and concludes that it fails because Fishman uses a small number of batches. From Monte Carlo experiments, Kleijnen suggests the use of at least 100 batches. The method, as proposed by Kleijnen (1987), is described in Section 2.

### H.2. TEST BASED ON THE VON NEUMANN STATISTIC.

Instead of using the usual autocorrelation estimator $\rho_1$ which has bias and shows a high standard error Kleijnen recommends the use of the "VON NEUMANN statistic (say q):

$$q = \frac{\sum_{i=1}^{n-1} (x_i - x_{i+1})^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \tag{H.1}$$

The q statistic reflects the first-order autocorrelation $\rho_1$. If the variables are independent (so that $\rho_1=\rho_2=...\rho_n=0$) then it can be proved that

$$E\ (q\,|\,\text{independent}\ \ x) = 2 \qquad \text{(H.2)}$$

whatever the distribution of x. If the variables show first-order autocorrelation ($\rho_1=0$) and the x are normally distributed then

$$E(q) = 2 - 2/n - 2E(\rho_1) \qquad \text{(H.3)}$$

We wish to know whether the observed value of q deviates significantly from the value 2 displayed in Equation H.2. Therefore we have to know the distribution of q. When the variables x are normally and independently distributed then it can be proved that

$$\sigma^2(q|x\text{~}NID) = \frac{4(n-2)}{(n-1)(n+1)} \qquad \text{(H.4)}$$

and then the distribution of q is approximately normal for n>20. Consequently we reject the hypothesis of independence if the observed q is smaller than 2 - $z^\alpha\ \sigma(q)$" ($z^\alpha$ is the upper point $\alpha$ of the normal distribution), "where $\sigma(q)$ follows from equation H.4. Based on analysis and Monte Carlo experiments we recommend using at least 100 subruns when testing the independence of the subrun responses." (Kleijnen, 1988)

### H.3. SOME OTHER TESTS FOR CORRELATION.

In a paper presented at the 1988 Winter Simulation Conference, Schmidt and Ho propose a new method for testing serial dependence among the batch means. As Kleijnen (1987), they conclude that Fishman's "method might not perform well if the sample sequence is too positively autocorrelated". This section describes the method proposed by Schmidt and Ho to deal with the problem of correlation.

"Schmidt and Ho have proposed a method (1987) of sequential systematic sampling to solve the problem of data correlation. Similar to replication and batch means, sequential systematic sampling also employs uncorrelated observations to assist construction of inferential procedures. Nevertheless, its sampling procedure could be viewed as the converse of batch means. While batch means groups a sequence of consecutive observations (one batch) together, sequential systematic sampling collects observations at intervals of some length (say k observations). If the correlation of sample sequence dies out at lag k, then observations drawn at intervals of k can be considered as essentially uncorrelated. Using a common value k as the batch size and the sampling interval, Ho and Schmidt (1987) conducted a simulation study of comparing batch means and sequential systematic sampling. The comparison is based upon the predictability of confidence interval procedures applied to sample observations generated from autoregressive, simple moving average and M/M/1 queuing models." (Schmidt, and Ho, 1988). The problem of estimating a value for k is discussed in the same paper. The basic problem is that of determining when the correlation has died out and it is approached in two phases. "In the first phase we propose a correlation estimate. Then we determine when the correlation has died out according to the criterion of Gross and Harris (1974). Hence a value k can be determined if the correlation at lag k is less than a small number. In phase 2 the Von Neumann ratio test is used to test whether observations drawn at intervals of k are uncorrelated." Schmidt, and Ho (1988)

# APPENDIX I : ADDITIONAL RESULTS FOR CHAPTER 6

## I.1. INTRODUCTION

This appendix contains empirical results obtained for some simulation models and that are used to illustrate the method proposed in Chapter 6, based on **weighted averages**, for the elimination of the influence of the initial conditions. In section I.1.1. we discuss very briefly the idea behind the new method and in section I.1.2. we discuss the measures of performance used in the evaluation of the performance of the new method as compared to methods where there is no elimination of the initial conditions or where this elimination is based on the deletion of some of the initial observations (run-in-period).

### I.1.1. WEIGHTED AVERAGES AND THEIR USE IN SIMULATION

The basic idea behind the **weighted averages** method is to assign a smaller weight to the initial values so as to minimise their influence on the estimate $X_j$ in replication *j*. In practice this estimate has been calculated from the simulation as the weighted average of $T_i$ observations (j=1, 2...k) as given by equation (I.1) and where $T_1$ is the first observation that is recorded, $T_2$ the second and so on.

$$X_j = \frac{\sum_{i=1}^{k} w_i * T_i}{\sum_{m=1}^{k} w_m} \tag{I.1}$$

In Equation I.1. $w_i$ is the weight assigned to observation *i*. In order to eliminate the influence of the initial conditions, $w_1 < w_2 < w_3 ... < w_k$.

One of the characteristics of this method is that the exact value of the weights is not extremely crucial. In other words, in some experiments we used:

$$w_i = i \tag{I.2}$$

while in others we chose exponential values for the weights as shown in Equation I.3.

$$w_i = (1 - e^{-k2*i}) \tag{I.3.}$$

with k2=0.0001, and also smaller or larger values. Experiments using different values for k2 were made and this value is not critical in the performance of the new method.

### I.1.2. MEASURES OF PERFORMANCE

The problem of the initialisation bias is due to the influence of the initial conditions. When the simulation is started "empty" and "idle", as it usually is, the initial observations are not representative of the steady state values. If the simulation run length is not long enough for this influence to disappear, the mean estimates obtained from the simulation will be biased. When a method to deal with the initialisation bias problem is used, the simulation run length required for the parameters to reach the steady state is considerably reduced. The results of the simulation will then be available sooner, and the extra computer time, that has been saved when the initialisation bias problem has been dealt with, can be used to take more replications. This will reduce the standard deviation of the mean estimate as well as giving a more accurate estimate.

For the evaluation of the alternative method proposed for the elimination of the initialisation bias problem we will compare the run length required for the parameter to reach the steady state, using the two different methods: **standard** and **weighted averages**. Elimination of the influence of the initial conditions implies that the run length required to reach the steady state is shorter for the method proposed in this chapter.

NOTE : **standard** is the method commonly used in simulation. In this method no attempt is made at reducing the influence of the initial conditions.

## I.2. EMPIRICAL RESULTS.

In this section we discuss results obtained for the mean queuing time and the mean queue length of some of the simulation models previously used in this thesis for testing new procedures.

### I.2.1. THE LAUNDERETTE MODEL

We obtained results for the mean queuing time and the mean queue length of the WASHQ, the BIDLE, the WMIDLE, and the DRIER queues. The mean estimates for these queues obtained as a function of the simulation run length and of the method used (run-in-period, weighted averages or standard) are given in the following tables:

I.1. (BIDLE), I.2. (WASHQ), I.3. (WMIDLE) and I.4. (DRIER).

In order to compare the simulation run lengths required for a parameter to reach the steady state under the different methods (weighted averages, run-in-period and standard) we compare the mean estimates to the real steady state value calculated in Appendix C. We consider that a parameter is in the steady state if the mean estimates fall within 2.5% of $\mu$; the value of $\mu$ as well as the range of values within 2.5% of $\mu$ are the following:

====== QUEUING TIME VALUES ======

| Queue | Steady state | Range |
|---|---|---|
| WASHQ | 6.675 | [6.508 , 6,842] |
| BIDLE | 67.260 | [65.580 , 68.941] |
| WMIDLE | 12.780 | [12.460 , 13.099] |
| DRIER | 1.990 | [1.940 , 2.040] |

====== QUEUE LENGTH VALUES ======

| Queue | Steady state | Range |
|---|---|---|
| WASHQ | 0.835 | [0.814 , 0.856] |
| BIDLE | 8.413 | [8.203 , 8.623] |
| WMIDLE | 1.599 | [1.559 , 1.639] |
| DRIER | 0.249 | [0.243 , 0.255] |

| BIDLE mean queuing time estimates | | | | | | |
|---|---|---|---|---|---|---|
| Run Length | Standard | | Weighted | | Run-In-Period | |
| | Mean | Std.Dev. | Mean | Std.Dev. | Mean | Std.Dev. |
| 2000 | 70.797 | 13.353 | 67.987 | 15.636 | 68.573 | 16.593 |
| 3000 | 70.524 | 12.003 | 69.172 | 14.044 | 69.173 | 13.906 |
| 4000 | 70.030 | 10.652 | 68.889 | 12.637 | 68.999 | 11.951 |
| 5000 | 69.495 | 9.366 | 68.350 | 10.930 | 68.615 | 10.181 |
| 6000 | 69.041 | 8.584 | 67.857 | 9.750 | 68.275 | 9.200 |
| 7000 | 68.892 | 8.087 | 67.920 | 9.422 | 68.238 | 8.639 |
| 8000 | 68.761 | 7.899 | 67.925 | 9.649 | 68.192 | 8.426 |
| 9000 | 68.761 | 7.536 | 68.095 | 9.046 | 68.261 | 8.003 |
| 10000 | 68.702 | 6.924 | 68.107 | 7.898 | 68.249 | 7.293 |
| 11000 | 68.613 | 6.519 | 68.041 | 7.269 | 68.196 | 6.809 |
| 12000 | 68.339 | 6.498 | 67.610 | 7.688 | 67.945 | 6.755 |
| 13000 | 68.107 | 6.376 | 67.283 | 7.794 | 67.735 | 6.610 |
| BIDLE mean queue length estimates | | | | | | |
| Run Length | Standard | | Weighted | | Run-In-Period | |
| | Mean | Std.Dev. | Mean | Std.Dev. | Mean | Std.Dev. |
| 2000 | 8.739 | 1.247 | 8.412 | 1.537 | 8.429 | 1.604 |
| 3000 | 8.699 | 1.162 | 8.528 | 1.424 | 8.511 | 1.380 |
| 4000 | 8.648 | 1.046 | 8.503 | 1.280 | 8.506 | 1.197 |
| 5000 | 8.605 | 0.926 | 8.472 | 1.114 | 8.489 | 1.030 |
| 6000 | 8.571 | 0.850 | 8.448 | 0.978 | 8.473 | 0.929 |
| 7000 | 8.559 | 0.797 | 8.456 | 0.941 | 8.475 | 0.863 |
| 8000 | 8.546 | 0.777 | 8.452 | 0.967 | 8.472 | 0.837 |
| 9000 | 8.549 | 0.748 | 8.478 | 0.918 | 8.485 | 0.801 |
| 10000 | 8.548 | 0.689 | 8.489 | 0.795 | 8.490 | 0.731 |
| 11000 | 8.544 | 0.645 | 8.491 | 0.720 | 8.492 | 0.679 |
| 12000 | 8.517 | 0.644 | 8.445 | 0.767 | 8.468 | 0.673 |
| 13000 | 8.493 | 0.635 | 8.409 | 0.788 | 8.447 | 0.661 |

**Table I.1.** BIDLE mean queuing time and mean queue length estimates (and standard deviation) as a function of the simulation run length and of the method used for recording the data (Standard or weighted).

| WASHQ Mean queuing time estimates | | | | | | |
|---|---|---|---|---|---|---|
| Run Length | Standard | | Run-In-Period | | Weighted Averages | |
| | Mean | Std. Dev. | Mean | Std. Dev. | Mean | Std. Dev. |
| 500 | 4.547 | 5.061 | | | 5.346 | 6.593 |
| 1000 | 5.320 | 5.092 | 5.708 | 8.790 | 5.910 | 6.304 |
| 1500 | 5.691 | 5.089 | 6.100 | 7.088 | 6.173 | 6.292 |
| 2000 | 5.855 | 4.635 | 6.190 | 5.860 | 6.257 | 5.561 |
| 2500 | 5.914 | 4.287 | 6.202 | 4.810 | 6.240 | 5.175 |
| 3000 | 5.976 | 4.140 | 6.203 | 4.810 | 6.265 | 5.116 |
| 4000 | 6.122 | 3.859 | 6.307 | 4.342 | 6.418 | 5.027 |
| 5000 | 6.246 | 3.891 | 6.406 | 4.279 | 6.457 | 5.011 |
| 6000 | 6.361 | 3.661 | <u>6.504</u> | 3.969 | <u>6.557</u> | 4.497 |
| 7000 | 6.424 | 3.373 | 6.550 | 3.619 | 6.612 | 3.984 |
| 8000 | 6.441 | 3.178 | 6.552 | 3.378 | 6.606 | 3.722 |
| 9000 | <u>6.488</u> | 3.016 | 6.590 | 3.186 | 6.663 | 3.578 |
| 10000 | 6.520 | 2.911 | 6.612 | 3.061 | 6.682 | 3.383 |
| 11000 | 6.517 | 2.796 | 6.601 | 2.929 | 6.641 | 3.233 |
| 12000 | 6.533 | 2.650 | 6.610 | 2.766 | 6.657 | 3.039 |
| WASHQ Mean queue length estimates | | | | | | |
| Run Length | Standard | | Run-In-Period | | Weighted Averages | |
| | Mean | Std. Dev. | Mean | Std. Dev. | Mean | Std. Dev. |
| 500 | 0.637 | 0.796 | | | 0.777 | 1.070 |
| 1000 | 0.712 | 0.772 | 0.808 | 1.274 | <u>0.811</u> | 0.977 |
| 1500 | 0.746 | 0.745 | 0.812 | 1.029 | 0.822 | 0.917 |
| 2000 | 0.759 | 0.658 | 0.806 | 0.8020 | 0.820 | 0.787 |
| 2500 | 0.763 | 0.608 | 0.795 | 0.665 | 0.815 | 0.746 |
| 3000 | 0.766 | 0.578 | 0.796 | 0.665 | 0.810 | 0.717 |
| 4000 | 0.781 | 0.533 | 0.804 | 0.597 | 0.826 | 0.709 |
| 5000 | 0.794 | 0.532 | <u>0.814</u> | 0.583 | 0.827 | 0.696 |
| 6000 | 0.807 | 0.493 | 0.824 | 0.533 | 0.837 | 0.609 |
| 7000 | <u>0.813</u> | 0.452 | 0.828 | 0.484 | 0.841 | 0.537 |
| 8000 | 0.814 | 0.424 | 0.827 | 0.450 | 0.839 | 0.504 |
| 9000 | 0.819 | 0.402 | 0.832 | 0.424 | 0.845 | 0.482 |
| 10000 | 0.822 | 0.387 | 0.834 | 0.406 | 0.847 | 0.456 |
| 11000 | 0.821 | 0.370 | 0.831 | 0.387 | 0.840 | 0.434 |

**Table I.2.** WASHQ mean queuing time and mean queue length estimates (and standard deviation) as a function of the simulation run length and of the method used for recording the data (standard or weighted).

| WMIDLE Mean queuing time estimates | | | | | | |
|---|---|---|---|---|---|---|
| Run Length | Standard | | Run-In-Period | | Weighted Averages | |
| | Mean | Std. Dev. | Mean | Std. Dev. | Mean | Std. Dev. |
| 500 | 15.646 | 7.113 | 14.134 | 8.181 | 13.671 | 7.804 |
| 1000 | 14.253 | 5.043 | 13.433 | 5.382 | 13.120 | 5.746 |
| 1500 | 13.837 | 4.202 | 13.282 | 4.376 | 13.070 | 4.804 |
| 2000 | 13.535 | 3.624 | 13.109 | 3.728 | 12.902 | 4.114 |
| 2500 | 13.381 | 3.300 | 13.038 | 3.372 | 12.856 | 3.748 |
| 3000 | 13.341 | 3.012 | 13.056 | 3.076 | 12.946 | 3.453 |
| 4000 | 13.255 | 2.637 | 13.041 | 2.678 | 12.975 | 3.073 |
| 5000 | 13.177 | 2.405 | 13.005 | 2.436 | 12.948 | 2.793 |
| 6000 | 13.122 | 2.184 | 12.978 | 2.206 | 12.929 | 2.537 |
| 7000 | 13.059 | 2.041 | 12.935 | 2.057 | 12.864 | 2.387 |
| 8000 | 13.031 | 1.940 | 12.923 | 1.951 | 12.854 | 2.271 |
| 9000 | 12.984 | 1.816 | 12.887 | 1.827 | 12.806 | 2.102 |
| 10000 | 12.958 | 1.713 | 12.871 | 1.720 | 12.789 | 1.989 |
| WMIDLE Mean queue length estimates | | | | | | |
| Run Length | Standard | | Run-In-Period | | Weighted Averages | |
| | Mean | Std. Dev. | Mean | Std. Dev. | Mean | Std. Dev. |
| 500 | 1.896 | 0.636 | 1.620 | 0.733 | 1.608 | 0.735 |
| 1000 | 1.752 | 0.476 | 1.614 | 0.510 | 1.599 | 0.561 |
| 1500 | 1.707 | 0.402 | 1.615 | 0.421 | 1.604 | 0.469 |
| 2000 | 1.677 | 0.349 | 1.608 | 0.361 | 1.596 | 0.405 |
| 2500 | 1.661 | 0.321 | 1.605 | 0.330 | 1.594 | 0.373 |
| 3000 | 1.657 | 0.295 | 1.611 | 0.302 | 1.605 | 0.344 |
| 4000 | 1.647 | 0.259 | 1.612 | 0.280 | 1.609 | 0.306 |
| 5000 | 1.639 | 0.239 | 1.611 | 0.256 | 1.608 | 0.282 |
| 6000 | 1.633 | 0.218 | 1.610 | 0.232 | 1.607 | 0.255 |
| 7000 | 1.627 | 0.203 | 1.606 | 0.214 | 1.602 | 0.238 |
| 8000 | 1.624 | 0.192 | 1.606 | 0.201 | 1.601 | 0.227 |
| 9000 | 1.619 | 0.181 | 1.603 | 0.188 | 1.597 | 0.211 |
| 10000 | 1.617 | 0.171 | 1.602 | 0.177 | 1.596 | 0.200 |

**Table I.3.** WMIDLE mean queuing time and mean queue length estimates as a function of the simulation run length and of the method used for recording the data (Standard or weighted).

| DRIER Mean queuing time estimates | | | | | | |
|---|---|---|---|---|---|---|
| Run Length | Standard | | Run-In-Period | | Weighted Averages | |
| | Mean | Std. Dev. | Mean | Std. Dev. | Mean | Std. Dev. |
| 500 | 4.846 | 2.215 | 2.700 | 2.146 | 2.611 | 2.192 |
| 1000 | 3.345 | 1.445 | 2.328 | 1.425 | 2.220 | 1.530 |
| 1500 | 2.893 | 1.183 | 2.229 | 1.174 | 2.146 | 1.295 |
| 2000 | 2.649 | 1.023 | 2.155 | 1.015 | 2.064 | 1.128 |
| 2500 | 2.513 | 0.920 | 2.120 | 0.915 | 2.032 | 1.005 |
| 3000 | 2.432 | 0.838 | 2.105 | 0.835 | 2.034 | 0.922 |
| 4000 | 2.337 | 0.721 | 2.093 | 0.718 | 2.059 | 0.818 |
| 5000 | 2.269 | 0.644 | 2.074 | 0.642 | 2.042 | 0.732 |
| 6000 | 2.228 | 0.587 | 2.066 | 0.585 | 2.036 | 0.674 |
| 7000 | 2.193 | 0.548 | 2.055 | 0.546 | 2.021 | 0.637 |
| 8000 | 2.166 | 0.515 | 2.045 | 0.513 | 2.012 | 0.598 |
| 9000 | 2.141 | 0.486 | 2.034 | 0.484 | 1.998 | 0.556 |
| 10000 | 2.125 | 0.453 | 2.028 | 0.451 | 1.995 | 0.521 |

**Table I.4.** DRIER mean queuing time estimates as a function of the simulation run length and of the method used for recording the data (Standard or weighted).

We do not give the mean queue length estimates for this queue as the steady state mean queue length value is very small, and as discussed in Section 2.6.2., we consider that in these cases a different estimator should be used.

To make it easier the comparison of the different methods, in each table, and for each case (weighted, run-in-period and standard) we have underlined the value for which the parameter can be considered to be already in the steady state. This means that the simulation run length corresponding to the underlined values is approximately the required run length for the parameter to reach the steady state.

From the different tables we can draw conclusions similar to those obtained for the PUB model in Chapter 6: with the use of the weighted averages of the individual observations, the parameters can be considered to reach the steady state for shorter simulation run lengths than when all the observations are assigned the same weight; the weighted averages method compared to the run-in-period method behaves just as well for the elimination of the influence of the initial conditions. There is some increase in the value of the standard deviation as compared to the value of the standard deviation of the estimate when a run-in-period is used. However, we can use the time that otherwise would have to be spent in the estimation of the run-in-period, to obtain more replications and run the simulation for a longer run length. This will give smaller estimates of the standard deviation.

### I.2.2. THE FISH PACKING MODEL.

Three queues have been studied in this model: the WAIT, the ALLQ and the IDLE queues. This model was simulated for two different combinations of the execution time of the activities that are given in Tables I.5.a and I.5.b.

| **Activity** | **Execution time** |
|---|---|
| ARRIVE | NEGEXP(7) |
| WEIGH | 4 |
| PACK | NORMAL(5,1) |

**Table I.5.a.** Execution time for the activities of the FISH PACKING model: Set 1.

| **Activity** | **Execution time** |
|---|---|
| ARRIVE | NEGEXP(6) |
| WEIGH | 2 |
| PACK | NORMAL(5,1) |

**Table I.5.b.** Execution time for the activities of the FISH PACKING model: Set 2.

Table I.6. gives the mean queuing time estimates for the WAIT, ALLQ and IDLE queues as a function of the simulation run length and of the method ("weighted" or "standard") used when the activity execution times are those given in Table I.5.a. Table I.7. gives the mean queuing time and the mean queue length for the ALLQ and the IDLE queues when the values of Table I.5.b. are used in the simulation.

We consider that a parameter is in the steady state when the mean estimates fall within 2.5% of the value of $\mu$, which has been obtained in an empirical way, as described in Chapter 2. The following are the values of $\mu$ as well as the range of values within 2.5% of $\mu$, for the conditions in Table I.5.a.:

| **Queue** | **Parameter** | **Value** | **Range** |
|---|---|---|---|
| WAIT | Queuing time | 2.694 | [2.626 , 2.761] |
| ALLQ | Queuing time | 5.166 | [5.037 , 5.295] |
| IDLE | Queuing time | 2.994 | [2.919 , 3.069] |

Steady state values for the values of Table I.5.b. are the following:

| **Queue** | **Parameter** | **Value** | **Range** |
|---|---|---|---|
| ALLQ | Queuing time | 14.900 | [14.257 , 15.272] |
| ALLQ | Queue length | 2.526 | [2.463 , 2.589] |
| IDLE | Queuing time | 4.020 | [3.919 , 4.120] |
| IDLE | Queue length | 0.666 | [0.649 , 0.683] |

In Tables I.6. and I.7. we have underlined those values for which the parameter can be considered to have already reached the steady state (i.e. beyond this estimate all fall within 2.5% of the steady state value). In this way we identify the required simulation run lengths for each parameter, and for each different method, to reach the steady state.

The conclusions from both tables are similar, and they show how when the method of **weighted averages** is used, the simulation run length required for the estimates to reach the steady state is shorter than when the standard method is used.

| FISH PACKING model - Mean queuing time estimates | | | | | | |
|---|---|---|---|---|---|---|
| | Weighted averages | | | Standard | | |
| Run Length | WAIT | ALLQ | IDLE | WAIT | ALLQ | IDLE |
| 500 | 2.655 | 4.670 | 2.989 | 2.561 | 4.097 | 3.074 |
| 1000 | 2.649 | 4.949 | 3.023 | 2.611 | 4.611 | 3.047 |
| 1500 | 2.664 | 5.008 | 2.999 | 2.637 | 4.786 | 3.023 |
| 2000 | 2.662 | 5.018 | 3.001 | 2.645 | 4.811 | 3.017 |
| 2500 | 2.672 | 5.047 | 3.001 | 2.655 | 4.865 | 3.013 |
| 3000 | 2.689 | 5.176 | 2.996 | 2.668 | 4.992 | 3.008 |
| 3500 | 2.708 | 5.198 | 3.000 | 2.682 | 5.045 | 3.008 |
| 4000 | 2.698 | 5.150 | 3.006 | 2.680 | 5.050 | 3.010 |
| 4500 | 2.696 | 5.087 | 3.008 | 2.681 | 5.026 | 3.010 |
| 5000 | 2.706 | 5.110 | 3.006 | 2.689 | 5.042 | 3.009 |
| 5500 | 2.701 | 5.094 | 3.010 | 2.688 | 5.041 | 3.010 |
| 6000 | 2.699 | 5.082 | 3.013 | 2.688 | 5.033 | 3.012 |
| 6500 | 2.694 | 5.075 | 3.010 | 2.687 | 5.038 | 3.010 |

**Table I.6.** Mean queuing time and mean queue length estimates for some queues of the FISH PACKING model for the conditions of Table I.5.a, as a function of the simulation run length and of the method of collecting the data.

| FISH PACKING MODEL - Mean queue length estimates | | | | |
|---|---|---|---|---|
| | Weighted averages | | Standard | |
| Run Length | ALLQ | IDLE | ALLQ | IDLE |
| 500 | 2.281 | 0.659 | 1.915 | 0.664 |
| 1000 | 2.483 | 0.663 | 2.212 | 0.665 |
| 1500 | 2.536 | 0.664 | 2.324 | 0.666 |
| 2000 | 2.517 | 0.665 | 2.350 | 0.666 |
| 2500 | 2.554 | 0.665 | 2.403 | 0.666 |
| 3000 | 2.570 | 0.665 | 2.438 | 0.666 |
| 3500 | 2.539 | 0.666 | 2.446 | 0.666 |
| 4000 | 2.509 | 0.666 | 2.439 | 0.666 |
| 4500 | 2.509 | 0.666 | 2.446 | 0.666 |
| 5000 | 2.501 | 0.666 | 2.445 | 0.667 |
| 5500 | 2.503 | 0.666 | 2.447 | 0.667 |
| 6000 | 2.509 | 0.666 | 2.455 | 0.667 |
| 6500 | 2.530 | 0.666 | 2.468 | 0.666 |
| 7000 | 2.542 | 0.666 | 2.477 | 0.666 |
| 7500 | 2.544 | 0.666 | 2.482 | 0.666 |
| Mean queuing time estimates | | | | |
| | Weighted averages | | Standard | |
| Run Length | ALLQ | IDLE | ALLQ | IDLE |
| 500 | 12.321 | 3.976 | 10.696 | 4.007 |
| 1000 | 13.817 | 3.997 | 12.600 | 4.005 |
| 1500 | 14.337 | 3.993 | 13.367 | 3.999 |
| 2000 | 14.323 | 3.995 | 13.568 | 3.999 |
| 2500 | 14.611 | 3.993 | 13.923 | 3.997 |
| 3000 | 14.775 | 3.996 | 14.169 | 3.998 |
| 3500 | 14.697 | 4.002 | 14.260 | 4.000 |
| 4000 | 14.551 | 4.001 | 14.241 | 4.000 |
| 4500 | 14.581 | 4.005 | 14.304 | 4.002 |
| 5000 | 14.567 | 4.007 | 14.320 | 4.004 |
| 5500 | 14.593 | 4.005 | 14.344 | 4.003 |
| 6000 | 14.628 | 4.003 | 14.390 | 4.002 |
| 6500 | 14.743 | 3.999 | 14.465 | 4.000 |
| 7000 | 14.831 | 3.998 | 14.526 | 3.999 |
| 7500 | 14.852 | 3.998 | 14.563 | 3.999 |
| 8000 | 14.852 | 3.996 | 14.586 | 3.998 |
| 8500 | 14.834 | 3.999 | 14.592 | 4.000 |

**Table I.7.** Mean queuing time and mean queue length for some of the queues of the FISH PACKING MODEL and the values given in Table I.5.b. as a function of the simulation run length and of the method: "Standard" or "Weighted" used in the simulation.

## I.2.3. THE STEELWORKS MODEL

Estimates for the queuing time and the queue length of the TBLOWQ and the PITQ queues were computed. In this simulation model we compare the **weighted averages** method with the run-in-period and with the **standard** method. The PITQ and the TBLOWQ mean queuing time estimates for the "Standard", "Weighted" and Run-In-Period methods as a function of the simulation run length are given in Table I.8.

| STEELWORKS - Mean queuing time estimates | | | | | | |
|---|---|---|---|---|---|---|
| Run Length | PITQ | | | TBLOWQ | | |
| | Standard | Weighted | Run-In-P | Standard | Weighted | Run-In-P. |
| 6000 | 30.6383 | 33.8627 | 36.6799 | 105.1568 | 94.1509 | 86.0791 |
| 10000 | 33.6059 | 36.4875 | 37.258 | 96.2046 | 86.7813 | 84.6342 |
| 15000 | 35.2066 | <u>37.5367</u> | <u>37.6838</u> | 91.461 | 83.9873 | <u>83.5879</u> |
| 20000 | 36.0177 | 37.9179 | 37.7791 | 88.9819 | <u>82.8771</u> | 83.3334 |
| 25000 | 36.4746 | 38.0452 | 37.9419 | 87.5213 | 82.4236 | 82.7845 |
| 30000 | 36.7769 | 38.1152 | 37.9952 | 86.6247 | 82.3156 | 82.6745 |
| 35000 | 37.0183 | 38.1998 | 38.0113 | 85.9084 | 82.1263 | 82.677 |
| 40000 | 37.2447 | 38.3348 | 38.0652 | 85.23 | 81.747 | 82.5283 |
| 45000 | 37.3185 | 38.2483 | 38.0727 | 84.9483 | 81.9353 | 82.4617 |
| 50000 | <u>37.4559</u> | 38.3264 | 38.1899 | 84.5231 | 81.7024 | 82.144 |
| 55000 | 37.533 | 38.3195 | 38.1802 | 84.2993 | 81.7637 | 82.1859 |
| 60000 | 37.6301 | 38.3758 | 38.2331 | 84.0088 | 81.6095 | 82.0328 |
| 65000 | 37.7108 | 38.4156 | 38.2535 | 83.7519 | 81.4743 | 81.9636 |
| 70000 | 37.7751 | 38.44 | 38.2935 | <u>83.5714</u> | 81.4367 | 81.8657 |
| 75000 | 37.8829 | 38.551 | 38.3794 | 83.2392 | 81.0803 | 81.604 |

**Table I.8.** TBLOWQ and PITQ mean queuing time estimates as a function of the simulation run length and of the method of dealing with the initialisation bias problem.

As in the previous tables, the underlined values correspond to the simulation run length for which the parameter can be said to have reached the steady state. A parameter will be considered to be in the steady state if the mean estimates fall within 2.5% of the steady state values (calculated in Appendix C); these values are the following:

| Queue | Steady state ($\mu$) | Range |
|---|---|---|
| TBLOWQ | 81.588 | [79.548 , 83.628] |
| PITQ | 38.405 | [37.445 , 39.365] |

As expected, using the weighted averages method the parameters reach the steady state for shorter simulation run lengths: for the PITQ for example, the mean estimates will reach this state for a simulation run length of 15000; if the standard method is used, a simulation run length longer than 50000 is required. The analysis of the results for the TBLOWQ queuing time is similar, and again shows that the simulation run length required for the parameter to reach the steady state is shorter, and sometimes by a significative amount of time, if the weighted averages method is used.

Table I.9. gives the standard deviation corresponding to the mean queuing and the mean queue length of the queues studied in this model, for the **standard**, the **weighted averages** and the **run-in-period** methods.

As in the case of the LAUNDERETTE model, the standard deviation of the weighted average estimate tends to be slightly larger than that when a run-in-period is used. But the difference between the two values is not so large as to make this new method useless due to its high variance. However, due to the simplicity of the weighted averages method and to the fact that it is not always possible to estimate the "optimal" run-in-period , we believe that the new method performs quite well as compared to the run-in-period method.

| STEELWORKS - standard deviation of the mean estimates. | | | | | | |
|---|---|---|---|---|---|---|
| | TBLOWQ mean queuing time | | | TBLOWQ mean queue length | | |
| Run length | Standard | Run-In | Weighted | Standard | Run-In | Weighted |
| 15000 | 16.040 | 20.647 | 18.876 | 0.532 | 0.686 | 0.629 |
| 20000 | 14.184 | 16.789 | 17.095 | 0.471 | 0.560 | 0.571 |
| 25000 | 13.034 | 14.955 | 15.974 | 0.434 | 0.499 | 0.533 |
| 30000 | 11.385 | 12.694 | 13.641 | 0.378 | 0.423 | 0.457 |
| 35000 | 10.498 | 11.540 | 12.379 | 0.349 | 0.385 | 0.415 |
| 40000 | 9.924 | 10.835 | 11.970 | 0.330 | 0.361 | 0.401 |
| 45000 | 9.454 | 10.168 | 11.110 | 0.314 | 0.339 | 0.372 |
| 50000 | 9.004 | 9.532 | 10.390 | 0.300 | 0.318 | 0.347 |
| 55000 | 8.336 | 8.803 | 10.211 | 0.277 | 0.293 | 0.341 |
| 60000 | 8.090 | 8.490 | 10.214 | 0.269 | 0.283 | 0.341 |
| 65000 | 7.861 | 8.257 | 10.453 | 0.262 | 0.275 | 0.348 |
| | PITQ mean queuing time | | | PITQ mean queue length | | |
| Run length | Standard | Run-In | Weighted | Standard | Run-In | Weighted |
| 15000 | 5.655 | 7.253 | 6.712 | 0.531 | 0.684 | 0.633 |
| 20000 | 5.010 | 5.933 | 6.061 | 0.472 | 0.561 | 0.574 |
| 25000 | 4.614 | 5.293 | 5.672 | 0.435 | 0.501 | 0.536 |
| 30000 | 4.019 | 4.488 | 4.876 | 0.379 | 0.424 | 0.459 |
| 35000 | 3.700 | 4.077 | 4.423 | 0.350 | 0.386 | 0.418 |
| 40000 | 3.506 | 3.832 | 4.272 | 0.331 | 0.362 | 0.404 |
| 45000 | 3.332 | 3.589 | 3.963 | 0.315 | 0.340 | 0.374 |
| 50000 | 3.186 | 3.374 | 3.667 | 0.300 | 0.318 | 0.347 |
| 55000 | 2.940 | 3.104 | 3.602 | 0.277 | 0.293 | 0.341 |
| 60000 | 2.849 | 2.993 | 3.599 | 0.269 | 0.283 | 0.341 |
| 65000 | 2.771 | 2.913 | 3.660 | 0.262 | 0.275 | 0.347 |

**Table I.9.** Standard deviation of the TBLOWQ and PITQ mean estimates as a function of the simulation run length.

### I.2.4. The M/M/4 QUEUE

In Appendix F we obtained the mean queuing time and mean queue length estimates for an M/M/4 queue with arrival rate $\lambda = 1/15$ and service rate $\mu = 1/50$. The steady state values are:

$W_q = 49.305$

$L_q = 3.287$

We have obtained mean queuing time and mean queue length estimates for this queue as a function of the simulation run length using the weighted averages method. These estimates are given in Table I.10. We have underlined the mean estimates for which the parameter can be considered to be in the steady state.

The conclusions from this table are similar to those of the previous cases considered in Chapter 6 and in this Appendix. When the weighted averages method is used, the mean estimates will fall within 2.5% of the steady state value for a shorter simulation run length than when the standard method is used.

### I.2.5. JACKSON'S MODEL

This model was described in Appendix A and results for the run-in-period estimated using the procedure proposed in Chapter 4 are given in Appendix F. In Table I.11. we give the mean estimates corresponding to the mean queuing time and the mean queue length of the queue of cars waiting for the ENGINE to be fitted, for both the **standard** and the **weighted averages** methods.

The steady state values ($\mu$) and the range of values within 2.5% of $\mu$ are the following:

| Steady state ($\mu$) | Range |
|---|---|
| 486.00 | [473.85 , 495.15] |
| 8.10 | [7.897 , 8.302] |

We have underlined the values beyond which the parameter can be considered to be in the steady state. The conclusion, as in all the previous examples is that the method of **weighted averages** performs better than the **standard** method and the mean estimates will reach the steady state for shorter simulation run lengths. While for a simulation run length of 130000 the mean estimates for the standard method are not yet within 2.5% of the steady state value, the weighted averages estimates reach the steady state for a simulation run length of 65000 for the mean queue length parameter.

| M/M/4 mean queuing time estimates | | | M/M/4 mean queue length estimates | |
|---|---|---|---|---|
| Run Length | Standard | Weighted | Standard | Weighted |
| 1500 | 29.824 | 35.195 | 2.118 | 2.566 |
| 2000 | 33.507 | 38.725 | 2.349 | 2.769 |
| 2500 | 35.965 | 40.865 | 2.499 | 2.892 |
| 3000 | 37.738 | 42.270 | 2.607 | 2.971 |
| 3500 | 38.961 | 43.402 | 2.684 | 3.039 |
| 4000 | 39.929 | 44.159 | 2.739 | 3.074 |
| 4500 | 40.901 | 45.035 | 2.799 | 3.131 |
| 5000 | 41.601 | 45.696 | 2.843 | 3.172 |
| 5500 | 42.197 | 46.273 | 2.876 | 3.199 |
| 6000 | 42.609 | 46.484 | 2.896 | 3.205 |
| 6500 | 43.198 | 47.148 | 2.930 | 3.240 |
| 7000 | 43.547 | 47.442 | 2.950 | 3.250 |
| 7500 | 43.873 | 47.590 | 2.967 | 3.251 |
| 8000 | 44.155 | 47.596 | 2.985 | 3.251 |
| 8500 | 44.419 | 47.593 | 2.999 | 3.242 |
| 9000 | 44.606 | 47.533 | 3.011 | 3.236 |
| 9500 | 44.833 | 47.580 | 3.025 | 3.238 |
| 10000 | 45.139 | 47.803 | 3.044 | 3.251 |
| 10500 | 45.376 | 47.990 | 3.058 | 3.261 |
| 11000 | 45.633 | 48.243 | 3.074 | 3.276 |
| 11500 | 45.903 | 48.522 | 3.090 | 3.291 |
| 12000 | 46.046 | 48.589 | 3.099 | 3.294 |
| 12500 | 46.184 | 48.587 | 3.107 | 3.293 |
| 13000 | 46.362 | 48.705 | 3.117 | 3.297 |
| 13500 | 46.514 | 48.791 | 3.126 | 3.301 |
| 14000 | 46.641 | 48.810 | 3.134 | 3.302 |
| 14500 | 46.763 | 48.835 | 3.142 | 3.303 |
| 15000 | 46.892 | 48.877 | 3.149 | 3.305 |
| 15500 | 47.016 | 48.968 | 3.158 | 3.313 |
| 16000 | 47.153 | 49.093 | 3.167 | 3.321 |
| 16500 | 47.293 | 49.226 | 3.175 | 3.328 |

**Table I.10.** M/M/4 mean queuing time and mean queue length estimates as a function of the simulation run length and of the method of obtaining the data.

| | Mean queuing time estimates | | Mean queue length estimates | |
|---|---|---|---|---|
| Run Length | Standard | Weighted | Standard | Weighted |
| 5000 | 217.727 | 254.676 | 3.778 | 4.557 |
| 10000 | 288.754 | 331.301 | 4.938 | 5.782 |
| 15000 | 329.325 | 372.133 | 5.607 | 6.453 |
| 20000 | 356.268 | 398.430 | 6.051 | 6.878 |
| 25000 | 376.750 | 417.976 | 6.380 | 7.183 |
| 30000 | 392.371 | 432.418 | 6.621 | 7.385 |
| 35000 | 402.754 | 439.852 | 6.782 | 7.486 |
| 40000 | 411.580 | 446.544 | 6.928 | 7.599 |
| 45000 | 419.991 | 454.307 | 7.064 | 7.719 |
| 50000 | 426.749 | 459.826 | 7.171 | 7.799 |
| 55000 | 431.832 | 463.095 | 7.251 | 7.845 |
| 60000 | 435.950 | 465.384 | 7.315 | 7.876 |
| 65000 | 440.236 | 468.737 | 7.387 | 7.932 |
| 70000 | 444.396 | 472.456 | 7.454 | 7.989 |
| 75000 | 448.146 | 475.736 | 7.515 | 8.041 |
| 80000 | 451.179 | 477.871 | 7.565 | 8.075 |
| 85000 | 453.508 | 478.981 | 7.600 | 8.085 |
| 90000 | 455.855 | 480.518 | 7.643 | 8.116 |
| 95000 | 458.082 | 482.047 | 7.680 | 8.141 |
| 100000 | 460.053 | 483.300 | 7.713 | 8.161 |
| 105000 | 462.525 | 485.728 | 7.751 | 8.194 |
| 110000 | 464.993 | 488.251 | 7.790 | 8.232 |
| 115000 | 466.054 | 488.114 | 7.803 | 8.220 |
| 120000 | 466.265 | 486.568 | 7.807 | 8.193 |
| 125000 | 466.323 | 484.953 | 7.806 | 8.163 |
| 130000 | 466.872 | 484.488 | 7.816 | 8.155 |

**Table I.11.** JACKSON'S model mean queuing time and mean queue length estimates as a function of the simulation run length and of the method for collecting the data.

Similarly, the weighted averages mean queuing time estimates reach the steady state for a simulation run length of 75000. In this particular case, the weighted averages method gives better results than the run-in-period method. As discussed in Appendix F, systems with a large value of the traffic intensity tend to have a large variance and standard deviation of the mean estimates, and in these cases, a large number of replications may be required for the estimation of the run-in-period.

# APPENDIX J : PASCAL MODIFICATIONS TO USE THE METHOD OF WEIGHTED AVERAGES

## J.1. INTRODUCTION

This Appendix explains how to modify a simulation software so that the method of **weighted averages** can be used for dealing with the problem of the initialisation bias problem. The simulation software used is VS6 and the code has been written in PASCAL.

In Section J.2. we explain the code used to record the queuing time parameter for a given queue, and in Section J.3. we discuss the procedure used to record the queue length of a queue.

## J.2. QUEUING TIME PROCEDURE

The procedure in the **standard** method is called "record_que_qtime" and it is called every time that an entity is removed from a queue. The variable "qent^.worknum"records the simulated time at which the entity was added to the queue. "TIM" is a variable used to represent the simulated time. Therefore, the line:

```
stay := TIM - qent^.worknum;
```

represents the time that the unit being removed from the queue has spent in the queue. The value of the queuing time returned by the program at the end of the simulation is the average of these individual queuing times recorded by the variable "stay". To keep track of these values we simply add them to a variable called "total", and the number of values (i.e. the number of times the procedure is called) added are recorded in the variable "count". Both variables have been initialised to "0" at the beginning of the simulation. When the variable TIM reaches the value corresponding to the total simulation run length, the mean queuing time is calculated as :

```
total/count
```

and this value is the value returned by VS6 as the mean queuing time corresponding to a single simulation run. If the method of replications is used we will change the random number seeds in each replications and therefore, different values of the mean queuing time are obtained corresponding to different replications.

For the **weighted averages** method (see corresponding procedure) we have underlined the two lines of the procedure that are modified to take into account the different "weights". The variable TOTAL1 is used along with the variable to change the "weights" of the individual observations. The variable TOTAL1 is initialised to 0 at the beginning of the simulation and is increased by 1 every time that the procedure is called. It can be seen then that the first time that the procedure is called, the observation recorded has a value of 0 for the queuing time. The second time the procedure is called the variable "weight" takes a value of

weight :=1-EXP(-(0.000001*TOTAL1)) (total1=1 now)

= 0.000001; and the variable total1 is increased again by 1;

Therefore, the second observation has a small weight, and as in the **standard** procedure, these observations are accumulated in the variable TOTAL. At the end of the simulation run length, the value returned by VS6 is the average of the different observations recorded, but each observation has different weight.

However, because of the way the method of **weighted averages** is used in forecasting the average is not given now by

total/count

but by

total/TOTAL2 , where TOTAL2 is the variable where the "weights" are accumulated.

**STANDARD PROCEDURE**

```
procedure record_que_qtime(h : que_histogram; qent : mod_ent);
var stay:longint; i:integer;
```

```
begin
  with h^do
  begin
    stay := TIM - qent^.worknum;
    cell[i] := cell[i]+1;
    total   := total + stay;
    sosq    := sosq + stay * stay;
    count   := count + 1;
  end;
end;
```

**WEIGHTED AVERAGES PROCEDURE**

```
procedure record_que_qtime(h  : que_histogram;  qent : mod_ent);
var stay:longint;  i:integer;weight:real;
begin
  with h^do
  begin
    stay := TIM - qent^.worknum;
    cell[i] := cell[i]+1;
    weight  :=1-EXP(-(0.000001*TOTAL1));
    total   := total + weight*stay;
    TOTAL1:=TOTAL1+1;
    TOTAL2  :=TOTAL2+weight;
    count  :=count+1;
  end;
end;
```

## J.3. QUEUE LENGTH PROCEDURE

The procedure called "LogQueData" is similar to that one of the queuing time, except that for each observation that is accumulated (i.e., every time the procedure is called) we consider not only the queue length at the particular moment of time that we call the procedure, but also the queue length for each unit of time. In other words, when call the procedure we assign the value of the simulated time at that particular time to the variable "tflag". We use this variable to count the number of units of time during which the length of the queue is not changed. This is shown in the following two lines:

val := qsize(que);

d := TIM-tflag;

Therefore, the value of the variable "d" is set equal to the number of units of time from the last call of the procedure, and the variable "val" evaluates the number of units in the queue at the simulated time that we call the procedure. This value has remained unchanged for a period of "d" units of time, and this value is accumulated in the variable "total" that keeps track of the accumulated queue length. At the same time, in the variable "count" is not increased by 1 as in the queuing time procedure but by "val" which is the number of individual queue lengths that we are adding to the accumulated length in the variable "total". This accumulated queue length considers the length at each unit of time. When the simulation is finished, the average queue length is calculated as before by

total/count;

The **standard procedure** is modified in the **weighted averages** procedure below, and the lines that are changed are underlined and are the following:

weight1:=1-EXP(-(0.000001*total1));

total := total + WEIGHT1*val * d;

The modification is similar to the one of the queuing time parameter. The "total" variable which represents the accumulated queue length is increased by a the queue length recorded by "val" with a "weight" given by "weight1", that changes every time that we call this procedure. In the variable "TOTAL2" we keep record of the "weights" that are used. They are accumulated and used to obtain the average queue length at the end of the simulation. This average queue length is now given by:

total/TOTAL2.

**STANDARD PROCEDURE**

```
procedure LogQueData(que:queue; hnam:str10);
var d,val:longint; i:integer; qhist:que_histogram;
begin
  qhist := que^.hist;
```

```
  with qhist^do
  begin
    val := qsize(que);
    d      := TIM-tflag;
    cell[i] := cell[i] + d;
    total   := total + val * d;
    sosq    := sosq + val * val * d;
    count   := count + d;
    tflag   := TIM;
  end;
 end;
```

**WEIGHTED AVERAGES PROCEDURE**

```
procedure LogQueData(que:queue;  hnam:str10);
var d,val:longint; i:integer; qhist:que_histogram;weight1:real;
begin
  qhist := que^.hist;
  with qhist^do
  begin
    val := qsize(que);
    d      := TIM-tflag;
    cell[i] := cell[i] + d;
    weight1:=1-EXP(-(0.000001*total1));
    total   := total + WEIGHT1*val * d;
    TOTAL2  :=TOTAL2+weight1*d;
    TOTAL1  :=TOTAL1+1;
    count   := count + d;
    tflag   := TIM;
    end;
end;
```

## REFERENCES

Adam N. (1983) Achieving a Confidence Interval for Parameters estimated by Simulation. *Management Science*, Vol 29, No. 7, July 1983.

Adlakha, V.G. and Fishman, G. (1982) Starting and Stopping rules for simulations using a priori information. *European Journal of Operational Research*. 379-394.

Amemiya, T. (1973) Generalised Least Squares with an estimated Autocovariance Matrix. *Econometrica*. 41, 723-732.

Amer, P.D. (1982) Providing Statistical Confidence in the evaluation of alternatives. *Simulation*, 38 (1982), 1 (Jan), pp. 13-19.

Andrews, R.W. and Schriber, T.J. (1983) A bayesian Batch Means Methodology for Analysis of Simulation Output. *Proceedings of the 1983 Winter Simulation Conference, 37-38.*

Andrews, D.F., Bickel, P.J., Hampel, F.R., Huber, P.J., Rogers, W.H., Tukey, J.W. (1972) *Robust Estimates of Location*, Princeton University Press, Princeton.

Anscombe, F.J. (1953) Sequential Estimation. *Journal Royal Statistical Society*, Series B, 50, No.1 : 1-29

Avramidis, A. and Wilson, J.R. (1989) A flexible method for estimating Inverse Distribution Functions in Simulation Experiments. SMS 89-21. Department of Statistics. School of Industrial Engineering. Purdue University.

Ayani, R.A. (1989) A parallel simulation scheme based on the distance between objects. *Proceedings of the SCS Multiconference on Distributed Simulation*, 21, 2 (March, 1989), 113-118.

Ayani, R. and Rajaei H. (1990) Parallel simulation of a generalised cube multistage interconnection network. *Proceedings of the SCS Multiconference on Distributed Simulation*, 22, 1 (January, 1990), 60-63.

Balmer, D.W. and Paul, R.J. (1986) CASM : The Right Environment for Simulation. *J. of the Opl. Res. Soc.*, 37, 443-452.

Banks, J. and Carson, J. (1984) Discrete-Event Simulation. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

Biles, W.E. (1975) A Response Surface Method for Experimental Optimization if Multiresponse Processes. *Industrial and Engineering Chemistry: Process Design and Development*, 14, 152-158.

Blomqvist, N. (1970) On the Transient Behaviour of the G1/G/1 Waiting Times. *Skand. Aktuarietidskr*, 118-129.

Box, G.E.P. and Draper N.R. (1959) A Basis for the Selection of the Response Surface Design. *Journal of the American Statistical Association*, 54, 42-52.

Brightman, H.J. (1978) Optimization through Experimentation: Applying Response Surface Methodology. *Decision Sciences*, 9, 481-495.

Carrol, W.C. (1961) The Created Response Surface Technique for Optimizing Restrained Systems. *Operations Research*, 9, 169-184.

Chan, T.F. and Lewis, J.G. (1979) Computing Standard Deviations: Accuracy. *Communications, ACM,* 22, No. 9, 526-531.

Chandy, K.M. and Sherman, R. (1989) The conditional event approach to distributed simulation. *Proceedings of the SCS Multiconference on Distributed Simulation,* 21, 2 (March, 1989), pp. 93-99.

Chen, Boy-Chun and Sargent, R.G. (1990) Confidence Interval Estimation for the Variance Parameter of Stationary Processes. *Management Science,* Vol. 36, 200-211.

Cheng, R.C.H. (1982) The Use of Antithetic Variates in Computer Simulation. *J. Opl. Res. Soc.,* 33, 229-237.

Cheng, R.C.H. (1984) Antithetic Variate Methods for Simulations of Processes with Peaks and Troughs. *EJOR (Netherlands)* Vol. 15, 227-236.

Cheng, R.C.H. and Davenport, T. (1988) Stratified Sampling in the Simplex with Applications to Estimating Statistical Distributions. *Proceedings of the 1988 Winter Simulation Conference,* 540-545.

Chew, Sew Tee. (1986) Program Generators for Discrete Event Digital Simulation Modelling. Unpublished Ph.D. thesis, University of London.

Chien, Chia-Hon (1988) Small-Sample Theory for Steady State Confidence Intervals. *Proceedings of the 1983 Winter Simulation Conference, 408-412.*

Choobineh, F. and Ballard, J.L. (1989) A Method of Confidence Interval Construction for Simulation Output Analysis. *Operations Research Letters,* North-Holland, Vol.8, 265-270.

Chow Y.S. and Robbins, H. (1965) On the Asymptotic Theory of Fixed-Width Sequential Confidence Intervals for the Mean. *Ann. Math. Statist.* Vol. 36, 457-462.

Churchill, Winston. Speech to the House of Commons, November 11, 1947.

Cinlar, E. (1975) An Introduction to Stochastic Processes. *Prentice-Hall,* 1975.

Clark, G.M. (1960) The utility of Statistics of Random Numbers. *Opns. Res.,* Vol. 8, 185-195.

Clark, G.M. (1980) Recursive Estimation of the variance of the sample average. *ACM Transactions on Mathematical Software.* 6, No.1, 58-67.

Cochran, J.K. and Cheng, Chuen-Sheng (1990) Automating the Procedure for Analysing Univariate Statistics in Computer Simulation Contexts. *Transactions of the Society for Computer Simulation,* Vol. 6, No. 3, 173-188.

COLLINS English Dictionary (1986), 2nd. Edition, William Collins Sons & Co. Ltd. Glasgow.

Cooley, B.J. and Houck, E.C. (1982) A Variance Reduction Strategy for the RSM Simulation Studies. *Decision Sciences,* 13, 303-321.

Cox,D.R. and Smith, W.L. (1961). Queues, Methan, London.

Crane, M.A. and Iglehart, D.L. (1974a) Simulating Stable Stochastic Systems; I. General Multiserver Queues. *J, Assoc. Comput. Mach.* 21, 110-113.

Crane, M.A. and Iglehart, D.L. (1974b) Simulating Stable Stochastic Systems; II. Markov Chains. *J. Assoc. Comput. Mach.* 21, 114-123.

Crane, M.A. and Iglehart, D.L. (1975a) Simulating Stable Stochastic Systems, III: Regenerative Processes and Discrete Event Simulation. *Operations Research*, 23, 33-45.

Crane, M.A. and Iglehart D.L. (1975b) Simulating Stable Stochastic Systems, IV: Approximation Techniques. *Management Science*, 21, 1215-1224.

Crane, M.A. and Lemoine, A.J. (1977) An Introduction to the Regenerative Method for Simulation Analysis. *Springer-Verlag, New York, N.Y.*

Crookes, J.G. and Valentine, B. (1982) Simulation in Microcomputers. *J. Opl. Res. Soc.* 33 855-858.

Crookes, J.G., Balmer, D.W., Chew, S.T. and Paul, R.J. (1986) A three phase Simulation System written in Pascal. *J. Opl. Res. Soc.*, 37, pp. 603-618.

DeBrota, D.J., Roberts,S.D., Dittus,R.S., and Wilson, J.R. (1988) Visual Interactive Fitting of Bounded Johnson Distributions Research Memorandum No. 88-3. Industrial Engineering, Purdue University.

DeBrota, D.J., Roberts,S.D., Dittus,R.S., and Wilson, J.R., Swain, J.J. and Venkatraman, S. (1989) Modelling Input Processes with Johnson Distributions. Report No. SMS 89-22. Department of Statistics, School of Industrial Engineering, Purdue University.

Deligonul, Z.S. (1987) Antithetic Bias Reduction for Discrete Event Simulations *J. Opl. Res. Soc.* Vol 38, No. 5, 431-437.

Deutsch, S.J., Richards, J.E., and Fernandez-Torres, C. (1983) The Initial Transient Problem: Estimating Transient and Steady State Performance. *Proceedings of the 1983 Winter Simulation Conference. 345-346.*

De Vries, R.C. (1990) Reducing null messages in Misra's distributed discrete event simulation method. *IEEE Transactions on Software Engineering*, 16, 1 (January 1990,), pp. 82-91.

Duersch, R. R. and Schruben L.W. (1985) Some Experiments with a run length control for simulations on PC's. *Annual Simulation Symposium.*

Duket, S.D. and Pritsker, A.A.B. (1978) Examination of Simulation Output using Spectral Methods. *Math. Comput. Simulation,* **20**, 53-60.

Efron, B.(1981) Nonparametric Standard Errors and Confidence Intervals. *The Canadian Journal of Statistics*, Vol 9, No.2, 139-172.

El Sheik, A.A.R., Paul, R.J., Harding, A.S. and Balmer, D.W. (1987) A Microcomputer based Simulation Study of a Port. *J. Opl. Res. Soc.*, **37** (8), 673-681.

Fieller, E.C. (1954) Some Problems on Interval Estimation. *J. Royal Statist. Soc.* Series B, No.2, 175-185.

Fishman, G.S. (1969) *Spectral Methods in Econometrics.* Harvard University Press, Cambridge, Mass.

Fishman, G.S. (1971) Estimating Sample Size in Computer Simulation. *Management Science,* **18**, 21-38.

Fishman, G. (1972) Bias Considerations in Simulation Experiments. *Operations Research,* **20** 785-790

Fishman, G.S. (1973) Concepts and Methods in Discrete Event Digital Simulation, John Wiley & Sons.

Fishman, G.S. (1974) Estimation in Multiserver Queueing Simulations. *Operations Research, 22*, 72-78.

Fishman, G.S. (1976) Batch Means in Digital Simulation. Technical Report 75-7, curriculum in Operations Research and Systems Analysis, University of North Carolina.

Fishman, G.S. (1978a) Grouping Observations in Digital Simulation. *Management Science, 24* 510-521.

Fishman, G.S. (1978b) *Principles of Discrete Event Simulation*, Wiley, New York.

Fox, B., Goldmsan, D. and Swain, J.J. (1991) Spaced Batch Means. *Operations Research Letters. North-Holland*, Vol. 10, 255-263.

Fraser, D.A.S. (1957) *NonParametric Methods in Statistics.* Wiley, New York.

Gafarian, A.V., Ancker, C.J. Jr., and Morisaku, F. (1978) Evaluation of Commonly used Rules used for Detecting "Steady State" in Computer Simulation. *Naval Research Logistics Quarterly*, 25, 511-529.

Glynn, P.W. (1982) Asymptotic Theory for NonParametric Confidence Intervals, Technical Report 63, Dept. of Operations Research, Stanford University.

Glynn, P. W. (1988) Simulation methods for queues: an overview. *Queueing Systems* (Switzerland), 3 (1988), 3 (July), pp. 221-256.

Glynn, P.W. and Iglehart, D.L. (1987) A New Initial Bias Deletion Rule. *Proceedings of the 1987 Winter Simulation Conference.* 318-319.

Glynn, P.W. and Iglehart, D.L. (1988) A new class of strongly consistent variance estimators for steady-state simulations. *Stochastic Processes and Applications*, 28, 1, 71-80.

Glynn, P.W. and Iglehart, D.L. (1990) Simulation Output Analysis using Standardised Time Series. *Mathematics of Operational Research*, 15, 1-16.

Goldsman, D. and Withers, D. (1990) Single-Replication Simulation. *Proceedings of the 1990 Winter Simulation Conference*. 387-391.

Goldsman, D. and Schruben, L. (1990) New Confidence Interval Estimators using Standardized Time Series. *Management Science*, Vol. 36, 393-397.

Gordon, C.C. (1969) System Simulation. Prentice-Hall, Englewood Cliffs, New Jersey, 1969.

Grassman, W.K. (1982) Initial Bias and Estimation Error in Discrete Event Simulation. *Proceedings of the 1982 Winter Simulation Conference*. 377-384.

Grassman, W.K. (1986) "Is the fact that the emperor wears no clothes a subject worthy of publication". *Interfaces*, 16, No.2, 43-51.

Grassman, W.K. (1988) Finding the Right Number of Servers in Real-World Queueing Systems, *Interfaces*, 18, No. 2, 94-103.

Gross, D. and Harris, C.M. (1974) Fundamentals of Queuing Theory, Wiley, New York, 1974.

Hall, P. (1983) Inverting an Edgeworth Expansion. *Ann. Statis.* 11, 569-576.

HARVARD GRAPHICS 3.0, Users Manual (1991), Software Publishing Corp., Mountain View, Calif., 1991.

Heidelberg, P. and Welch, P.D. (1981a) A Spectral Method for Confidence Interval Generation and Run Length Control in Simulations. *Comm. Assoc. Comput. Mach,* 24, 233-245.

Heidelberg, P. and Welch, P.D, (1981b) Adaptive Spectral Methods for Simulation Output Analysis. *IBM Journal of Research and Development,* 25, 860-876.

Heidelberg, P. and Welch P. (1983) Simulation Run Length Control in the presence of an Initial Transient. *Operations Research,* Vol 31, No. 6, November-December.

Heikes, R.G.D., Montgomery, D.C., and Rardin, R.L. (1976) Using Common Random Numbers in Simulation Experiments: An approach to Statistical Analysis. *Simulation,* 25 81-85.

Ho, C.F. and Schmidt, J.W. (1987) An Investigation of Statistical Parametric Inferential Procedures with Correlated Sample Observations. *Proceedings of the Pan Pacific Conference IV,* 818-823.

Holder, R.D., Gittins, R.P., and Burke, D.M. (1989) The Effects of Warship and Replenishment Ship Attrition on War Arsenal Requirements. *J. Opl. Res. Soc.,* 40, No.2, 167-175.

Hollocks, B. (1984) The impact of the Microcomputer in Simulation. *Operational Research '84 (Netherlands),* 901-907.

Iglehart, D.L. (1975) Simulating Stable Stochastic Systems; V. Comparison of Ratio Estimators, *Naval Res. Logist. Quart.,* Vol. 22, 553-565.

Iglehart, D.L. and Stone, M.L. (1983) Regenerative Simulation for Estimating Extreme Values. *Operations Research,* Vol. 31, No.6, 1145-1166.

Jackson, J. (1957) Networks of Waiting Lines. *Operations Research,* Vol. 5, 518-521.

James, B.A.P. (1985) Variance Reduction Techniques. *J. Opl. Res. Soc.* 36, No. 6, 525-530.

Johnston, J., *Econometric Methods*, 2nd. Ed., McGraw-Hill, New York, 1972.

Kabak, I.W. (1968) Stopping Rules for Queuing Simulations. *Operations Research,* Vol. 16, 431-437.

Karlin, S. and Taylor, H. (1975) A First Course in Stochastic Processes. Academic Press. 1975

Kelton, W.D. (1980) The Startup Problem in Discrete Event Simulation, Technical Report 80-1. Dept. of Industrial Engineering, University of Wisconsin.

Kelton, W.D. (1982) Initial Transient Techniques for Stochastic Simulation, Technical Report WS-8204. Dept. of Administrative Sciences, Kent. State University.

Kelton, W.D. (1983) Simulation Analysis. *Proceedings of the 1983 Winter Simulation Conference,* 159-168.

Kelton, W.D. (1985) Analysis of Simulation Output Data. *Proceedings of the 1985 Winter Simulation Conference,* 33-34.

Kelton, W.D, and Law, A. M. (1983) A New Approach for Dealing with the Startup Problem in Discrete Event Simulation. *Naval Research Logistics Quarterly.* Vol. 30, pp 641-658.

Kelton W.D. and Law, A.M. (1984) An analytical Evaluation of Alternative Strategies in Steady State Simulation. *Operations Research,* 32 169-184.

Kelton, W.D. and Law, A. (1985) The Transient Behaviour of the M/M/s Queue with Implications for Steady-State Simulation. *Operations Research,* Vol.33, No.2, 378-396.

Kimbler, D.L. and Knight, B.D. (1985) A Survey of Current Methods for the Elimination of Initialisation Bias in Digital Simulation. *Annual Simulation Symposium.*

Kleijnen, J.P.C. (1979) Analysis of Simulation with Common Random Numbers: A note on Heikes et al (1976) Version 2. *Katholieke Hogeschool Fac. Econ. Wet. tech. Rep.* 79.095 Tilburg, Netherlands.

Kleijnen, J.P.C., Van Der Ven, R. and Sanders, B. (1982) Testing Independence of Simulation Subruns: A Note on the Power of the Von Neumann Test. *Eur. J. Opnl. Res.* 9, 92-93

Kleijnen, J.P.C. (1987) Statistical Tools for Simulation Practitioners. Marcel Dekker, Inc. New York.

Kvanli, Alan H. (1988) STATISTICS : A Computer Integrated Approach. West Publishing Company. St. Paul, Minn.

Lavenberg, S.S. and Welch, P.D. (1981) A perspective on the use of Control Variates to increase the efficiency of Monte Carlo Simulations. *Management Science,* 27, 322-335

Law, A. (1977) Confidence Intervals in Discrete Event Simulations: A comparison of Replications and Batch Means. *Naval Research Logistics Quarterly. Vol.24, 667-678.*

Law, A. (1980) Statistical Analysis of the Output Data from Terminating Simulations. *Naval Research Logistics Quarterly, Vol.27, 131-143.*

Law, A. (1982) Practical Statistical Analysis of Simulation Output Data: The State of the Art. *Proceedings of the 1982 Winter Simulation Conference,* 641-642.

Law, A. (1983) Statistical Analysis of Simulation Output Data. *Operations Research,* Vol 31., No. 6, 983-1029.

Law, A.M. (1990) Design and Analysis of Simulation Experiments for Manufacturing Applications. *Proceedings of the 1990 Winter Simulation Conference.* Osman Balci, Randall P. Sadowski, Richard E. Nance (Eds).

Law, A. and Carson, John S. (1979) A Sequential Procedure for Determining the Length of a Steady State Simulation. *Operations Research,* Vol 27, No.5, September, October 1979

Law, A., Kelton, W.D., and Koenig, L.W. (1981) Relative Width Sequential Confidence Intervals for the Mean. *Commun. Statist.* **B10,** 29-39.

Law, A, and Kelton, W.D. (1982a) Confidence Intervals for Steady State Simulations. II. A Survey of Sequential Procedures. *Management Science,* Vol. 28, No 5, May 1982.

Law, A.M. and Kelton, W.D. (1982b) Simulation Modelling and Analysis. McGraw-Hill, Inc.

Law, A.M. and Kelton, W.D. (1991) Simulation Modelling and Analysis, 2nd. Edition. McGraw-Hill, Inc.

Law, A.M. and Kelton, W.D. (1984) Confidence Intervals for Steady State Simulations: I. A Survey of Fixed Sample Size Procedures. *Operations Research*, Vol. 72, No. 6.

Lewis, P.A.W. and Orav, E.J. (1989) Simulation Methodology for Statisticians, Operations Analysts, and Engineers. Volume 1. Wadsworth & Brooks. Belmont, Calif.

Lilliefors, H.W. (1967) On the Kolmogorov-Smirnov Tests for Normality with Mean and Variance Unknown. *Journal of the American Statistical Association.* 62, 399-402.

Lin, Y-B and Lazowska, E.D. (1989) Optimality considerations of "Time Warp" parallel simulation. *Tech. Rep. 89-10-06*, Dept. of Computer Science, University of Washington, Seattle, Washington, 1989.

Little, J.D.C. (1961) A Proof for the Queueing Formula $L = \lambda w$. *Operations Research*, Vol. 16, 651-665.

Madansky, A. (1976) Optimal Initial Conditions for a Simulation Problem. *Opns. Res.* 27, 572-577.

Mamrak, S.A. and Amer, P.D. (1980) Estimating Confidence Intervals for Simulations of Computer Systems. *Simulation* 35, No.6, 199-205.

Mechanic, H. and McKay, W. (1966) Confidence Intervals for Averages of Dependent Data in Simulations II. Technical Report 17-202, IBM Advanced Systems, Development Division.

Meketon, M.S. and Heidelberg, P. (1982) A Renewal Theoretic Approach to Bias Reduction in Regenerative Simulations. *Management Science*, Vol. 28, 173-181.

Meketon, M.S. and Schmeiser, B. (1984) Overlapping Batch Means: Something for Nothing? *Proceedings of the 1984 Winter Simulation Conference.* 227-230.

Minh, Do Le (1989) A variant of the conditional expectation variance reduction technique and its application to the simulation of the G1/G1/1 queues. *Management Science.* Vol. 35, No.11.

Mukhopadhyay, N. (1980) A consistent and asymptotically efficient two-stage procedure to construct fixed width confidence intervals for the mean. *Metrika,* 27, 281-284.

Newell, G.F. (1971) Applications of Queueing Theory. Chapman and Hall, London.

Noether, G.E. (1972) Distribution-Free confidence Intervals. *American Statistician.* 26, No.1, 39-41.

Paul, R.J. and Balmer, D.W. (1985). Simulation Lecture Notes. The London School of Economics and Political Science.

Paul, R.J. and Chew, S.T. (1987) Simulation Modelling Using an Interactive Simulation Program Generator. *J. Opl. Res. Soc.,* 38, 735-752.

Pidd, M. (1992) Computer Simulation in Management Science. 3rd. Edition. John Wiley & Sons.

Pritsker, A.A.B. (1984) Applications of Simulation. *Operational Research '84, IFORS,* 812-854.

Pritsker, A.A.B. (1989) Why Simulation works. *Proceedings of the 1989 Winter Simulation Conference.*

Reed, D.A., Malony, A.D. and McCredie, B.D. (1988) Parallel Discrete Event Simulation using Shared Memory. *IEEE Transactions Software Engineering*, 14,4 (April, 1988), pp. 541-553.

Ripley, B.D. (1988) Simulation methodology: An introduction for queueing theorists. *Queueing Systems*, 3 (1988) 3 (July), pp. 201-220.

Robbins, H., Simons, G. and Starr, N. (1967) A sequential analogue of the Behrens-Fisher problem. *Annals Mathematical Statistics*, 38, 1384-1391.

Robinson, D.W. (1976) Determination of Run Length in Simulations of Stable Stochastic Systems. Technical Report, 86-21, , Control Analysis Corporation, Palo Alto, Calif.

Royston, J.P. (1982) The W test for Normality. *Applied Statistics*, 31, 176-180.

Safizadeh, M.H. (1990) Optimization in Simulation: Current Issues and the Future Outlook. *Naval Research Logistics*, Vol. 37, 807-825.

Saliby, E. (1990a) Understanding the Variability of Simulation results: An empirical study. *J. Opl. Res. Soc.*, No. 4, pp 319-327,

Saliby, E. (1990b) Descriptive Sampling : A better Approach to Monte Carlo Simulation. *J. Opl. Res. Soc.*, Vol. 41, No. 0, pp 101-109.

Saliby, E. and Paul, R.J. (1992) Implementing Descriptive Sampling in Three Phase Discrete Event Simulation Models. Submitted for publication, *JORS*.

Sauer, C. (1979) Confidence Intervals for Queueing Simulations of Computer Systems. *Performance Evaluation Rev.*, 8, 45-55.

Schmeiser, B. (1982) Batch size effects in the Analysis of Simulation Output. *Operations Research*, Vol 30, No.3, May-June 1982.

Schmidt, J.W. and Ho, C.F. (1988) An algorithm for testing serial dependence of simulation output data. *Proceedings of the 1988 Winter Simulation Conference*, M. Abrams, P. Haigh and J.Comfort (eds).

Schriber, T.J. and Andrews, R.W. (1979) Interactive Analysis of Simulation Output by the Method of Batch Means. *Proceedings of the 1979 Winter Simulation Conference*. ACM, New York, N.Y.

Schriber, T.J. and Andrews, R.W. (1981) A Conceptual Framework for Research in the Analysis of Simulation Output. *Communications of the ACM*, Vol.24, No.4, 218-233.

Schruben, Lee W. (1982) Detecting Initialisation Bias in Simulation Output. *Operations Research*, Vol. 30, No. 3 May-June, 1982

Schruben, L.W. (1983) Confidence Interval Estimation using Standardised Time Series. *Operations Research*, Vol 31, No. 6, 1090-1107.

Schruben, L.W. and Margolin, B.H. (1978) Pseudo-random Number Assignment in Statistically designed Simulation and Distribution Sampling Experiments. *J. of the Amer. Statist. Assoc.*. Vol 73, No. 363, 504-525.

Schruben, L. and Singh H. (1983) Optimal tests for Initialisation Bias in Simulation Output. *Operations Research*, Vol. 31 No. 6, November-December, 1983.

Schruben L. and Cogliano, V.J. (1985) An Experimental Procedure for Simulation Response Surface Identification. *Technical Report No. 669, Cornell University*.

Schruben, L. and Yucesan, E. (1988) Transaction tagging in highly congested queueing simulations. *Queueing Systems (QUESTA)*, 3, 3, 257-264.

Schruben, L. and Jacobson, S.H. (1989) Techniques for Simulation Response Optimisation. *Operations Research Letters*, North-Holland, Vol. 8, 1-9.

Seila, A.F. (1990) Output Analysis for Simulation. *Proceedings of the 1990 Winter Simulation Conference.*

Shapiro, S.S. and Wilk, M.B. (1965) An Analysis of Variance Test for Normality (Complete Samples). *Biometrika*, Vol. 52, 591-611.

Sharon, A.P. and Nelson, B.L. (1988) Analytic and External Control Variates, *Journal of the Operational Research Society*, Vol.39, No.6, 595-602.

Snell, M. and Schruben L.W. (1979) Weighting Simulation Data to Reduce Initialisation Effects. Technical Report 395 School of Industrial Engineering, Cornell University.

Snell, M. and Schruben L.W. (1985) Weighting Simulation Data to Reduce Initialisation Effects. *IIE Transactions*, Vol. 17, No.4, 354-363.

Srivastava, M.S. (1970) On a sequential analogue of the Behrens-Fisher problem. *Journal Royal Statistical Society*, Series B, 32, No.1, 144-148.

Starr, N. (1966) The Performance of a Sequential Procedure for the Fixed-width Interval Estimation of the Mean. *Ann. Math. Statist.*, Vol. 37, 36-50.

STATGRAPHICS 4.0. (Statistical Graphics System) User's Manual, 1989

Stephens, M.A. (1974) EDF Statistics for Goodness-of-Fit and Some Comparisons. *Journal of the American Statistical Association*, Vol. 69, 730-737.

Symons, A. (1985) What is User Friendliness in Modelling and Simulation. *Annual Simulation Symposium.* 69-73.

Tin, M. (1965) Comparison of Some Ratio Measures. *Journal of the American Statistical Association,* Vol. 60, 294-307.

Turnquist, M.A. and Sussman, J.M. (1977) Toward Guidelines for Designing Experiments on Queueing Simulation. *Simulation,* 28, 137-144.

Wagner, D.B. and Lazowska, E.D. (1989) Parallel simulation of queueing networks: Limitations and potentials. *Proceedings of 1989 ACM SIGMETRICS and PERFORMANCE,* '89, 17,1 (May 1989), 146-155.

Welch, P.D. (1981) On the Problem of the Initial Transient in Steady State Simulation. *IBM Watson Research Centre, Yorktown Heights, N.Y.*

Welch, P.D. (1983) The Statistical Analysis of Simulation Results. In *The Computer Performance Modelling Handbook,* 268-328, S.S. Lavemberg (Ed) Academic Press, New York.

Whitt, W. (1989a) Planning Queueing Simulations. *Management Science.* 1341-1366.

Whitt, W. (1989b) Simulation Run Length Planning. *Proceedings of the 1989 Winter Simulation Conference,* 106-112.

Williams, T.M., Gittins, R.P. and Burke, D.M. (1989) Replenishment at Sea. *J. Opl. Res. Soc.,* 40, 10, 881-887.

Wilson, J.R. (1984) Variance Reduction Techniques for Digital Simulation. *American Journal of Mathematical and Management Science.* 4, 277-312.

Wilson, J.R. and Pritsker, A.A.B. (1978a) A Survey of Research on the Simulation Start Up Problem. *Simulation,* 31, 55-58.

Wilson, J.R. and Pritsker, A.A.B. (1978b) Evaluation of StartUp Policies in Simulation Experiments. *Simulation,* 31, 79-89.

Winston, W.L. (1987) Operations Research : Applications and Algorithms. Duxbury Press, Boston.

Withers, C.S. (1983) Expansions for the Distribution and Quantiles of a Regular Functional of the Empirical Distribution with Applications to Non-Parametric Confidence Intervals. *Ann. Statist.,* 11, 577-587.

Woolsey, G. (1979) Whatever happened to simple simulation? A question and an answer. *Interfaces,* 9, Vol. 4, 9-11.

Wright, R.D. and Ramsay, T.E. (1979) On the Effectiveness of Common Random Numbers. *Management Science,* 25 649-656.

# BIBLIOGRAPHY

## SIMULATION STUDIES

Abbott, R.A. and Greene, T.J. (1982) Determination of Appropriate Dynamic Slack Sequencing Rules for an Industrial Flow Shop Via Discrete Simulation. *Proceedings of the 1982 Winter Simulation Conference, 223-230.*

Alexander, E.L. Jr. (1976) A simulation Approach to Computer Performance Evaluation using GASP IV. *Proceedings Summer Computer Simulation Conference*, 6-10.

Auterio, V.J. (1974) Q-GERT Simulation of Air Terminal Cargo Facilities. *Proceedings Pittsburgh Modelling and Simulation Conference.* Vol.5, 1974, 1181-1186.

Bellas, C.J. and Samli, A.C. (1973) Improving New Product Planning with GERT Simulation. *California Management Review*, Vol XV, 1973, 14-21.

Bookbinder J. H. and Higginson J.K. (1986) Customer Service vs. Trim Waste in Corrugated Box Manufacture. *J. Opl. Res. Soc.* Vol 37, No.11 1061-1071.

Butler, J.K. (1982) Assessment of Test Protocol Using Discrete Simulation: The Antaneus Report. *Presented at 1982 Winter Simulation Conference.* 1982.

Davies, H. and Davies, R. (1987) A Simulation Model for Planning Services for Renal Patients in Europe. *J. Opl. Res. Soc.* Vol 38, No. 8, 693-700.

Duket, S. and Wortman, D. (1976) Q-GERT Model of the Dover Air Force Base Port Cargo Facilities. MACRO Task Force, Military Airlift Command, Scott Air Force Base, Illinois, 1976.

Federal Power Commission Exhibit EP-237, (1976) Risk Analysis of the Arctic Gas Pipeline Project Construction Schedule. Vol 167, Federal Power Commission, 1976.

Lyon, R.B. (1976) Radionuclide pathway Calculations Using a Set of Computer Programs Interfaced with GASP IV. *Proceedings of the 1976 Winter Simulation Conference.* 549-557.

Miner, R. and Pritsker, A.A.B. (1980) Dynamics Modelling: The ICAM Definition Language ($IDEF_2$) Approach. *Proceedings AIIE Fall Conference,* 1980.

Peart, R.M. et Al. (1985) Some Examples of the use of Simulation in U.S. and Canadian Agriculture. *J. Opl. Res. Soc.* Vol 36, No.2 109-115.

Pidd, M. (1987) Simulating Automated Food Plants. *J. Opl. Res. Soc.* Vol 38, No.8 683-692.

### GOODNESS-OF-FIT TESTS

Breiman, L. (1973) Statistics: With a View Towards Applications. *Houghton-Mifflin, Boston.*

Chernoff, H. and Lehman, E.L. (1954) The Use of Maximum Likelihood Estimates in $\chi^2$ Tests for Goodness-of-fit. *Ann. Math. Statist.,* Vol. 25, 579-586.

D'Agostino, R.B. (1971) An omnibus Test for Normality of Moderate and Large Size Samples. *Biometrika,* 58, 341-348.

D'Agostino, R.B. (1982) Small Sample Probability Points for the D Test of Normality, *Biometrika.* 59, 219, 221.

Durbin, J. (1975) Kolmogorov-Smirnov Tests when Parameters are Estimates with Applications to Tests of Exponentiality and Tests on Spacing. *Biometrika,* Vol. 62, 5-22.

Gail, M.H. and Gastwirth,J.L. (1975) A Scale-Free Goodness-of-fit Test for the Exponential Distribution Based on the Gini Statistic. *Journal of the Royal Statistical Society,* Series B, 40, 350-357.

Lewis, P.A.W. (1961) Distribution of the Anderson-Darling Statistic. *Annals of Mathematical Statistics,* 32, 1118-1124.

Lilliefors, H.W. (1969) On the Kolmogorov-Smirnov Test for the Exponential Distribution with Mean Unknown. *Journal of the American Statistical Association.* 64, 387-389.

Pettitt, A.N. and Stephens, M.A. (1977) The Kolmogorov-Smirnov Goodness-of-fit Statistic with Discrete and Grouped Data, *Technometrics,* Vol. 19, 205-210.

Shapiro, S.S., Wilk, M.B. and Chen, H.J. (1968) A Comparative Study of Various Tests for Normality. *Journal of the American Statistical Association.* Vol. 63, 1343-1372.

Shapiro, S.S. and Francia, R.S. (1972) An Approximate Analysis of Variance Test for Normality. *Journal of the American Statistical Association.* Vol. 67, 215-216.

Stephens M.A. (1970a) Kolmogorov-type tests for exponentiality when the scale parameter is unknown. Technical Report No. 154, Department of Statistics. Stanford University.

Stephens, M.A. (1970b) Use of Kolmogorov-Smirnov, Cramer-Von_Mises and Related Statistics without extensive tables. *Journal of the Royal Statistical Society,* Ser B. 32, No.1, 115-122.

Stephens, M.A. (1971) Asymptotic Results for Goodness-of-fit Tests with Unknown Parameters. Technical reports No. 159, 180. Department of Statistics. Stanford University.

Stephens, M.A. (1978) Goodness-of-Fit Tests with Special Reference to Tests for Exponentiality. *Stanford University, Department of Statistics.* Technical Report No. 22, Stanford, California.

Weisberg, S. (1974) An Empirical Comparison of the Percentage Points of W and W'. *Biometrika,* Vol. 61, 644-646.

Weisberg, S. and Bingham, C. (1975) An Approximate Analysis of Variance Test for Non-Normality Suitable for Machine Calculation. *Technometrics,* Vol. 17, 133-134.

## GENERATION OF RANDOM VARIABLES

Ahrens, J.H. and Dieter, U. (1972) Computer Methods for Sampling from the Exponential and Normal Distributions. *Communications of the ACM* Vol. 15, No. 10, 873-882.

Ahrens, J.H. and Dieter, U. (1974) Computer Methods for Sampling from Gamma, Beta, Poisson and Binomial Distributions. *Computing,* 12, 223-246.

Atkinson, A.C. (1979) A Family of Switching Algorithms for Computer Generation of Beta Random Variables, *Biometrika,* 66, 141-145.

Atkinson, A.C. and Whittaker, J. (1976a) A Switching Algorithm for the Generation of Beta Random Variables with at least One Parameter less than 1. *Applied Statistics*, 28, 90-93.

Atkinson, A.C. and Pearce, M.C. (1976b) The Computer Generation of Beta, Gamma, and Normal Random Variables. *Journal Royal Statistical Society*. Series A, 139, 431-448.

Box, G.E.P. and Muller, M.E. (1958) A Note on the Generation of Random Normal Deviates. *Ann. Math. Statist*, 29, 610-611.

Cheng, R.C.H. (1977) The Generation of Gamma Variables with Non-Integral Shape Parameter. *Applied Statistics*, 26, 71-75.

Cheng, R.C.H. (1978) Generating Beta Variates with NonIntegral Shape Parameters. *Communications A.C.M.*, 21, 317-322.

Cheng, R.C.H. and Feast, G.M. (1979) Some simple Gamma Variate Generators, *Applied Statistics*, 28, 290-295.

Filliben, J.J. (1975) The Probability Plot Correlation Coefficient Test for Normality. *Technometrics*, 17, 111-117.

Forsythe, G.E. (1972) Von Neumann's Comparison Method for Random Sampling from the Normal and Other Distributions. *Math. Comput.* 26, 817-826.

Kinderman, A.J. and Ramage, J.G. Computer Generation of Normal Random Variables. *Journal American Statistician Association*, 71,893-896.

Ramberg, J.S. and Schmeiser, B.W. (1972) An Approximate Method for Generating Symmetric Random Variables. *Communications of the ACM,* Vol. 15, No.11, 987-990.

Tadikamalla, P.R. (1978) Computer Generation of Gamma Random Variables-II. *Commun. Ass. Comput. Mach.*, 21, 925-928.

**BATCH MEANS ALTERNATIVE METHODS**

Conway, R.W. (1963) Some Tactical Problems in Digital Simulation. *Management Science.* Vol 10, 47-61.

Damerdji, H. (1989) Strong Consistency of the Variance Estimator in Steady State Simulation Output Analysis. *Technical Report, Department of Operational Research, Stanford University,* Stanford, Cal.

Romeu, J.L. (1987) Another Look at the Comparisons of the Non-overlapping Batch Means and Area Simulations Output Analysis Procedures. *Actas II Seminario Internacional de Investigacion Operativa en el Pais Vasco (Spain).* 161-187.

Schmeiser, B. (1990) Overlapping Batch Statistics. *Proceedings of the 1989 Winter Simulation Conference.* 395-398.

Welch, P.D. (1987) On the Relationship Between Batch Means, Overlapping Batch Means, and Spectral Estimation. *Proceedings of the 1987 Winter Simulation Conference.* 320-323.

**OUTLIERS DETECTION**

Andrews, D.F. and Pregibon, D. (1978) Finding the Outliers that matter. *Journal Royal Statistical Society,* Series B, 40, No.1, 85-93.

Barnett, V. and Lewis, T. (1978) Outliers in Statistical Data. John Wiley & Sons. New York, second edition 1985.

Beckman, R.J. and Cook, R.D. (1983) Outlier...s. (and discussion). *Technometrics,* 25, No.2, 119-163.

Brown, B.M. and Kildea, D.G. (1979) Outlier-detection tests and robust estimators based on signs of residuals. *Communications in Statistics, Theory and Methods,*. A8, No.3, 257-269.

Iglewicz, B. and Martinez, J. (1982) Outlier Detection Using Robust Measures of Scale. *Comput. Simul.* Vol. 15, 285-293.

Jain, R.B. et al (1982) A unified Approach for estimation and detection of Outliers. *Communications in Statistics, Theory and Methods.* 11, No. 25, 2953-2976.

Miyashita, B. and Newbold, P. (1983) On the Sensitivity to Nonnormality of a Tests for Outliers in Linear Models. *Communications in Statistics, Theory and Method,* 12, No.12, 1413-1419.

**OPTIMISATION IN SIMULATION**

Biles, W.E. (1975) A Response Surface Method for Experimental Optimization in Multiresponse Processes. *Industrial and Engineering Chemistry: Process Design and Development,* 14, 152-158.

Biles, W.E. (1977) Strategies for Optimization of Multiple Response Simulation Models. *Proceedings of the 1977 Winter Simulation Conference,* 135-142.

Curry, G.L. and Hartfiel, D.J.. (1983) A Simulation Optimization Method: Its Convergence and Utility. *Naval Research Quarterly*, 30, 227-236.

Daughety, A.F. and Turnquist, M.A. (1980) Simulation Optimization Using Response Surfaces based on Spline Approximations. *Proceedings of the 1980 Winter Simulation Conference,* 183-193.

Heller, N.B. and Staats, G.E. (1973) Response Surface Optimization When Experimental Factors are Subject to Costs and Constraints. *Technometrics.* 15, 113-123.

Leung, Ying Tat (1987) Single Run Optimization of Discrete Event Simulation- An empirical Study Using the M/M/1 Queue. *IIE Transactions*, Vol. 21, 35-48.

Montgomery, D.C. and Evans, D.M. (1975) Second Order Response Surface Designs in Computer Simulations. *Simulation*, 26, 113-121.

Myers, R.H. (1971) Response Surface Methodology, Allyn and Bacon, Boston.

Myers, R.H. and Carter, W.H. (1973) Response Surface Techniques for Dual Response Systems. *Technometrics*, 15, 301-317.

## VARIANCE REDUCTION TECHNIQUES

Nelson, B.L. (1987) On Control Variate Estimators. *Comput. Opns. Res.* 14 219-225.

Nelson, B.L. (1990) Control Variate Remedies. *Operations Research.* Vol.38, No. 6, 974-992.

Sharon, A.P. and Nelson, B.L. (1988) Analytic and External Control Variates. *J. Opl. Res. Soc,* Vol 39, No. 6, 595-602.

OTHERS

Cramer, H. (1946) *Mathematical Methods of Statistics.* Princeton, N.J. Princeton University Press.

Efron, B. (1987) Better Bootstrap Confidence Intervals. *J. Amer. Statist. Assoc.* 82, 171-185.

Fishman, G.S. (1976) Sampling from the Gamma Distribution on a Computer. *Comm. Assoc. Comput. Mach,* Vol. 19, 407-409.

Fishman, G.S. and Moore, L.R. (1978) Starting times for Data Collection in a queueing Simulation. I: Experiments with a single server model. Technical Report TR78-1. Curriculum in Operational Research and Systems Analysis. The University of North Carolina.

Fishman, G.S. (1977) Achieving Specific Accuracy in Simulation Output Analysis. *Comm. Assoc. Comput. Mach.* Vol. 20, 310-315.

Glynn, P.W. (1982a) Coverage Error for Confidence Intervals arising in Simulation Output Analysis. *Proceedings of the 1982 Winter Simulation Conference.* 369-375.

Glynn, P.W. (1982b) Asymptotic Theory for Nonparametric Confidence Intervals, Technical Report 59, Dept of Operations Research, Stanford University.

Glynn, P.W. (1983) On the Role of Generalised Semi-Markov Process in Simulation Output Analysis. *Proceedings of the 1983 Winter Simulation Conference.*

Goldsman, D. and Schruben L. (1984) Asymptotic Properties of Some Confidence Interval Estimators for Simulation Output. *Management Science*, Vol. 30, 1217-1225.

Goldsman, D. and Meketon, M.S. (1986) A Comparison of Several Variance Estimators. *Technical Report, J-85-12, School of Industrial and Systems Engineering, Georgia Institute of Technology*, Atlanta, GA.

Heidelberg, P. and Iglehart, D.L. (1979) Comparing Stochastic Systems using Regenerative Simulations with Common Random Numbers. *Adv. Appl. Prob.* 11 804-819

Johnson, N.J. (1978) Modified *t*-tests and Confidence Intervals for Asymmetrical Populations. *J. Amer. Statist. Assoc.* 73 536-544.

Kang, K. and Schmeiser, B. (1983) Evaluation and Comparison Methods for Confidence Interval Procedures. *Proceedings of the 1983 Winter Simulation Conference*, Research Summary.

Mihram, G.A. (1983) Simulation Methodology: Statistical Aspects. *Proceedings of the 1983 Winter Simulation Conference*, 27-35.

Murray, J.R. and Kelton, W.D. (1988) Initializing for Bias Reduction: Some Analytical Considerations. *Proceedings of the 1988 Winter Simulation Conference.* 546-548.

Nadas, A. (1969) An extension of a Theorem of Chow and Robbins on Sequential Confidence Intervals. *Ann. Math. Statist*, Vol 40, No. 2, 667-671.

Own, D.B. (1962) Handbook of Statistical Tables. *Addison-Wesley, Reading, Mass.*

Pearson, K. (1900) On a Criterion that a Given System of Deviations from the Probable in the Case of a Correlated System of Variables is such that it can be reasonably supposed to have arisen in Random Sampling. *Phil. Mag.* (5)50, 157-175.

Reiser, M. and Kobayashi H. (1975) Queueing Networks with Multiple Closed Chains: Theory and Computational Algorithms. IBM Journal of Research and Development, 19,3, May 1975.

Von Neumann, J. (1951) Various Techniques Used in Connection with Random Digits. *Nat. Bur. Std. Math. Ser,* 12, 36-28.

Wilson, J.R. (1984) Statistical Aspects of Simulation. *Operational Research '84.* IFORS 1984, 825-841.

Wonnacott, T.H. and Wonnacott, R.J. (1984). Introductory Statistics for Business and Economics. John Wiley & Sons.