Markovian modelling and optimisation for data analysis and queueing systems

N. A. Tankeh

A thesis submitted in
partial fulfilment for the degree of
Doctor of Philosophy of the University of London
and the Diploma of Membership of Imperial College

Department of Electrical Engineering
Imperial College of Science Technology and Medicine

November 10, 2002

BIBL. LONDN. UNIV.

Abstract

The following problem(with some modifications) is taken from actual industrial practice for data and system development. A central control station is designed to serve a number of subscribers with their demands. The characteristics of the demand process which set it off from other data analysis and queueing problems are that the sizes of the data workloads that are brought in by the primary "arrivals", the individual customers, are random variables having a general probability distribution rather than being exponentially distributed. Our problem follows. Firstly, given a finite time interval, what is the expected total demand made? Secondly, suppose that the demand is processed at constant rate (which is provided as a bandwidth measure of the link out of the station) whenever the inventory of the unprocessed demand is non–negative, what is the expected total unprocessed demand at the end of the finite time interval?

Two features complicate the solution to this prediction problems: (1) demands arrive at instances that are not necessarily Poisson and demand quantities brought in at these instances do not constitute a Markov process; (2) the workload prediction is required in a finite time and not just at the steady state as is usually considered in current research.

To solve the problems, we suppose that the number of customer "arrivals" during a finite time interval is a renewal random variable and the workload brought by each individual is also a renewal random variable. Assuming stochastic independence, the probability distribution of the total demands for all the subscribed customers during this finite time interval is a compound renewal process. Given data records of arrival instances and the sizes of the demand, the data analysis problem is to make prediction about future demands. We show that under certain conditions the compound renewal process can be approximated as a Gaussian time series. A neuro–fuzzy networks is used for data approximation and demand prediction. The modelling is formulated as a model reduction problem and optimisation methods are used to select the best neuro–fuzzy structures. This framework is extended to handle cases when the noise is correlated with non–stationary statistical properties.

Our case study also includes data communication switch which controls message passing among computing nodes using independent operating systems. Here, the inventory system is just a queue of messages that need to be transmitted from the memory of one computing node to the other. In the first transient $M/D/1/N$ model for message passing across Asynchronous Transfer Mode (ATM), it was envisaged that communication of data messages across computing nodes occur on fixed message sizes over a buffer communication channel. We have extended this idea to more general queues such as $M/G/1/N$, $G/G/1/N$, including $M/G/2/N$ queue with two channel adaptors per node.

Using IBM 32–way Regatta/SP switching technology, we judiciously tune parameters in our models to predict I/O and link bandwidths, prediction queue length packets in communication channels. We extended the modelling framework to a wide range of message passing protocols and provided advice that assist the research and

development of future protocols. This extension includes successful applications to the modelling of parallel file systems for remote data access in disks and zero copy data communication.

The additional advantage of this Markovian framework is that it provides a context for incorporating messages of various distributions, leading to analysis of communication protocol of different packet sizes. Another contribution is that we have developed a transient model in a discrete form, making it possible to use Markov chains analytical methods like generating functions.

Contents

1 Introduction **13**
1.1 The central problems 13
1.2 Insurance and electricity demand processes 21
1.3 Data communication across computing switch 22
1.3.1 Client and server clustering 24
1.4 Queueing model of data communication 25
1.4.1 Our strategy for $M/G/1/N$ queue 29
1.4.2 A new discrete PDP theory 29
1.5 Neuro–fuzzy applications to data analysis 30
1.5.1 Model reduction 30
1.5.2 Relationship to neural networks 34
1.5.3 Relationship to fuzzy logic 36
1.6 Our contribution 37
1.7 Notations used in the thesis 39

2 Model reduction and optimisation **41**
2.1 Introduction 41
2.1.1 Model reduction problem 44
2.2 Methods based on analysis of variance 45
2.3 Gaussian time series approximation 48
2.4 Computer communication among nodes 53
2.4.1 Batching for model reduction 55
2.4.2 Selecting important data points 58
2.4.3 Some results based on the analysis of variance 62
2.5 Orthogonal optimisation algorithms 63
2.5.1 Conjugate Gradient algorithms 66
2.5.2 Some results on optimisation 69
2.5.3 Comparing different RBFs 70
2.5.4 Results for truncated electricity data 72
2.5.5 Regression with daily claims data 74
2.6 Model reduction with constraint optimisation 75
2.6.1 Regularisation and Ridge regression 75
2.6.2 Application of F–Distribution 79
2.6.3 Confident interval for constraint least–squares 83

2.6.4 Experiments with constrained least squares 83
2.7 Summary 85

3 Modelling Correlation and Transient effects **87**
3.1 Introduction 87
3.1.1 Steady state in queue simulation 88
3.2 Correlated noise sequences 90
3.3 Our modelling framework 94
3.4 Estimating the correlation structure 96
3.5 Correlation functions 97
3.5.1 Partial correlation from Covariance matrix 98
3.5.2 Testing for heteroscedastic using ACF and PCF 99
3.6 Expirements on volatility 106
3.6.1 Experiment on finite horizon prediction 108
3.7 Experiments: Correlated errors 111
3.8 Experiment on regression using batch data 113
3.9 Further Experiment with autoregressive and correlated errors 115
3.9.1 Estimating mean from insurance daily claims 117
3.10 Experiment on model reduction 119
3.10.1 Information criteria 121
3.11 Summary 121

4 Optimization of covariance structures **123**
4.1 Introduction 123
4.2 Preliminaries 124
4.3 Semi-definite programming (SDP) with LMI 127
4.3.1 Softwares and applications 128
4.3.2 Why SDP methods 130
4.4 Factor analysis 130
4.5 Educational testing theory 131
4.6 Fletcher problem 132
4.7 Procrustes problem 132
4.8 Clustering with covariance matrix 133
4.9 Optimal design of training data matrix A^TA 137
4.9.1 Results using the tea data 140
4.9.2 Results using test of English as a Foreign language 143
4.9.3 Preference analysis with tea demand sample 144
4.9.4 Identification from noisy tea demand 146
4.10 Factor analysis on tea demand 150
4.11 Summary 155

5 Markov models for queues **157**
5.1 Introduction 157
5.2 Other methods of modeling M/G/1/N queues 160

5.2.1 Features of PDPs 161
5.2.2 Generators of PDPs 166
5.3 Expectations of PDPs 166
5.3.1 Infinite–time horizon 166
5.3.2 Finite horizon expectation 167
5.4 Generators of queueing processes 170
5.4.1 Dynamic of $M/G/1/N$ service process 172
5.4.2 Generator of M/G/1/N model 174
5.4.3 Generator two–server $M/G/2/N$ queue 177
5.4.4 Generator of $G/G/1/N$ queue 179
5.5 Summary 180

6 Computation and simulation of $M/G/1/N$ queues 182
6.1 Introduction 182
6.2 Numerical solution of $M/G/1/N$ queue 183
6.2.1 Modelling service distribution 187
6.3 Simulation of service systems 188
6.3.1 Computing the truncation point D 190
6.4 Computing hazard rates 192
6.4.1 Approximation of $M/D/1/N$ model of Davis 193
6.5 Simulation software 195
6.6 Simulation results of $M/G/1/N$ with Arena 197
6.6.1 SIMULIN–like flow diagram 200
6.7 Software integration 201
6.8 Results of $M/G/1/N$ using PDPs theory 201
6.8.1 Comparism with Morse Erlang model 202
6.9 Blocking probability 204
6.10 Demand and supply 206
6.11 Conclusion 213

7 Towards a discrete PDP model 223
7.1 Discrete PDP model 224
7.2 Generator of discrete renewal process 226
7.3 Discrete Renewal Theorem 227
7.4 Discrete approximation of Renewal Density 230
7.5 Discrete model of $M/D/1/N$ queue 233
7.5.1 Discrete model for $M/G/1/N$ queue 235
7.6 Markov chain models for queues 236
7.6.1 $M/D/1/N$ 236
7.6.2 $M/E_L/1/N$ 237
7.7 Conclusion 238

8 Data communication across an IBM switch 240
8.1 Introduction 240

8.2 Motivation 243
8.2.1 Parallel MPI application 243
8.2.2 Applications to parallel file systems 244
8.3 Major achievements in data communication 245
8.4 Review of I/O communication models 246
8.5 Software protocol model 249
8.6 Estimating distribution of message length 252
8.6.1 Computing the Harzard rate 255
8.7 PDP model for software protocol communication 256
8.8 Upwind scheme for protocol queue model 258
8.8.1 Results of packet length estimation 260
8.9 Estimating and optimising link bandwidth u 264
8.10 Conclusion 271

9 Conclusion **273**
9.1 Achievements 274
9.2 Future directions 276

10 Appendix: Data sets **278**
10.1 Introduction 278
10.1.1 Electricity consumption data sets 279
10.2 Insurance–claims 280
10.3 Preference scores: tea data set 281
10.4 Education test scores 284
10.4.1 Production yield over different batches 285
10.5 Product sale prices 285
10.6 House valuations in different neighbourhoods 285
10.7 Rubber consumption 286
10.8 Basic fact about testing F–hypothesis 287
10.8.1 Covariance and schur complements 288

11 References **291**

Bibliography **292**

List of Figures

1.1 Superposition of renewal processes 17
1.2 Clustering computers for message passing 23
1.3 Radial basis function 35

2.1 Gaussian approximation to a Binomial distribution 52
2.2 Replication mean across batches 56
2.3 Replication mean across batches 57
2.4 Replication mean across batches 58
2.5 The 4 fuzzy rules do not overlap 61
2.6 The 10 fuzzy rules overlap 62
2.7 Data points selection 63
2.8 $\beta = 120$: converges with 6 neurons 64
2.9 Evolution of mean square with Arnoldi algorithm. 71
2.10 Different membership functions for RBFs 72
2.11 Model reduction with residuals 73
2.12 Model reduction with residuals 74
2.13 Daily claims data 75
2.14 training data: claims data 76
2.15 Model reduction plot: Daily claims data 77
2.16 Effects of ridge parameters 78
2.17 Effects of ridge regression 80

3.1 Queue lengths vs replications 89
3.2 Batch mean 90
3.3 Batch average 91
3.4 Batch average 92
3.5 Electricity demand as Logarithmic series 101
3.6 ACF for the time series 102
3.7 PCF for the time series 103
3.8 ACF for the squared– time series 104
3.9 PCF for the squared– time series 105
3.10 Forecasting standard deviation 110
3.11 Plots 110
3.12 Forecasting standard deviation in the long range 111
3.13 Batching data 113

3.14 Batching data 114
3.15 Daily insurance claims 117
3.16 Log of daily insurance claims 119

4.1 Positive definite matrix cone 129
4.2 Clustering with minimum ellipsoid 139
4.3 Relationship in tea preference for $p = 2$ 145
4.4 Planar view of factor loadings 154
4.5 Rotation of factor loadings 154

5.1 The sample path of a PDP process 163
5.2 Cumulative investment in a single call 172

6.1 Inter arrival times of customers 189
6.2 Exponential distribtion 190
6.3 Erlang (2) distribtion 191
6.4 Erlang(5) distribution 192
6.5 Comparing Erlang (20) with normal 193
6.6 Hazard rate of Erlang(20) 194
6.7 Hazard rate of Erlang(20) 195
6.8 Hazard rate of Erlang(20) 196
6.9 Hazard rate of Erlang(20) 197
6.10 Que length with offer load $\rho = 0.9$ 198
6.11 Hazard rates of some Erlang distributions 199
6.12 Variation of queue length with Erlang k 200
6.13 Comparing PDP with Morse's results 202
6.14 Hitting probability 205
6.15 Davis queue 206
6.16 From Davis theory 207
6.17 Blocking probability 208
6.18 Idle probability 209
6.19 From Davis theory with D30 210
6.20 Davis theory with D10 211
6.21 Queue length from Davis theory with D20 212
6.22 Another truncation D 213
6.23 Effect of D 214
6.24 Truncation D 215
6.25 Effects of truncation D 215
6.26 More queue transient 216
6.27 Queue example 216
6.28 Queue example 217
6.29 Example queue 217
6.30 Queue illustration 218
6.31 Example queue 218
6.32 Queue with more speed of server 219

6.33 Effect of server speed 219
6.34 Another illustration of the effect of server speed 220
6.35 Queue with increase speed of server 220
6.36 Speed of Second server is twice that of first server 221
6.37 Speed of Second server is 3 times that of first server 221
6.38 An example queue 222
6.39 Speed of Second server is same as that of first server 222

8.1 A message passing setup between two Nodes 247
8.2 A smal example of a discrete distribution 255
8.3 Courant CFL condition 259
8.4 Number of packets in system(in micro seconds) 261
8.5 Number of packets in system(in micro seconds) 262
8.6 Number of packets in system(in micro seconds) 263
8.7 Number of packets in system(in micro seconds) 264
8.8 No. Packets in flight 265
8.9 DMA traffic paths for zero copy 266
8.10 Queue length with bandwidth 1.5GB/sec 269
8.11 Queue length with bandwidth 1.3GB/sec and buffer time 300 pclks . 270
8.12 Queue length with bandwidth 1.5GB/sec and buffer time 250 pclks . 271
8.13 Queue length with bandwidth 1.5GB/sec and buffer time 300 pclks . 272
8.14 Queue length against buffering delay and bandwidth 272

10.1 Electricity consumption 279
10.2 Daily claims over 7 years 281
10.3 Daily claims over 7 years 282
10.4 Chi–square distribution 288
10.5 Student distribution 289

Acknowledgements

I would like to express my sincere thank to my Supervisor, Prof. Mamdani (IC), for his constant guidance and invaluable advice throughout this project. I could do no less than express my deep gratitude to Dr. J. M. Howl(IC), Dr. P. Buah(IC/IBM), Dr. A, Dassios (LSE) and Dr. D. Williams (Oxford University). I shall always remember their valuable advice and helpful discussions.

The practical motivation for the problem brought to light arose from research works at BT laboratories in Ipswich, U.K., where I had the opportunity to exchange ideas with, N. Azarmi, Dr. B. Azvine, Dr. H. Nwana and Dr. D. Ndumu.

It would have been an omission not to mention Dr. Susan Pitts and Dr. Y. Liu of the University of Cambridge and Prof. B. Buxton in UCL for their constructive advice.

I am also very much indebted to Imperial College, University of Cambridge and University College London (UCL) which provided me with research fellowships that financially supported my research work.

Finally, I want to thank my family (Adela(wife), Charles(son), Lisa (daughter)) andMicheal(brother) for supporting me throughout my studies, particularly Charles and Lisa who kept their crying at nights to the minimum during the crucial stages of this work.

Chapter 1

Introduction

1.1 The central problems

This thesis is concerned with the modelling of a class of demand process whose behaviour is characterised by a super–position of some random jumps, the later occurring as a counting process. The research programme leading to this thesis started off by modelling discrete data recorded over time from a demand process. These data sets, outlined in the Appendix were obtained from inventory systems such as daily electricity consumption, daily insurance claims (including record of number of customers who made the claims), number of customers who arrive in a queueing system and the record of the queueing times under different service distributions. The *burning problems* are based on making predictions, finding appropriate model for a given set of observations and finding conditions under which a given model is a good approximation. This modelling project is further complicated because each data set of observation is neither necessarily stationary nor Markov, in the sense defined below. Furthermore, it is well–known that most counting processes are renewal processes, but it is not clear under which conditions these processes can be approximated by Gaussian processes; or if errors of approximation exhibit no apparent deviation from being statistically independent. How can we choose the best model for a given counting process?

The original goal was simply to develop algorithms that could be used to make a time dependent prediction of the level of demand. It was then proposed that such

processes could be modelled as a class of discrete non–linear dynamical systems, closely related to stochastic dynamical systems, where the theory of linear time series is a subset of this general class. These are state–space models where the present state can be explicitly represented as a non–linear function of past variables, plus some additive Gaussian noise. It turns out that this admits a Markovian realisation. This discrete structural representation does not imply that all discrete stochastic models are Markovian: it does indicate that if enough variables are incorporated in the definition of "*immediate past*", a forgetfulness, all but that past is a sufficient approximation, one that can help us make predictions about future states of the system.

Much of the interest in studying these non–linear models lies in the fact that it offers an approach in modelling dynamical systems using neural–fuzzy networks, systems that are inherently non–linear but feasible for numerical optimisation. These systems were very successful in the modelling and the prediction of a large number of these data sets.

The question that arose was whether this principle of constructing a state–space discrete Markov structure and then using a non–linear neuro–fuzzy system for the approximation could be generalised to other cases such as the prediction of queue length, expected number of customers or the amount of inventory in a general renewal process. Or, if not, whether an approach that meets with a degree of analytical explanation can be sought.

This thesis argues that non–linear modelling with discrete state space models does not generalise to all demand processes or general renewal processes, that those that can be modelled in this way have specific stochastic properties. Some neural networks researchers avoid these questions by considering the idealized discrete structure and carry on as if queue length is internally consistent with the model. In (Chen and Leslie 1992), for example, queue length at the present time is just a non–linear function of some past queue lengths. In other approaches (Habib 1995, Hiramatsu 1990), the same assumptions enter in one or another form, explicitly or tacitly, in identification of the object of modelling. No attempt is made to capture or formulate any concept that does not fall within the scope circumscribed by this approach, rooted on the assumption that the only uncertainty is Gaussian noise.

By facing the question of queue length, applying the non–linear discrete time systems using neuro–fuzzy representation, we found that the procedures suggested were seriously inadequate and in any event, could not possibly be understood (and were not intended) to provide an answer to the transient queue length, even in the narrow

conditions where most of these work were focused. As soon as we started to tackle the queue length issue with renewal theory, a wide range of new phenomena were discovered, including quite simple ones that had been ignored within the confine of machine learning community, those concerned with applications to demand for services.

This is what accounts for the research path taken, shift of focus, revisions of research plans, the challenges it faced and the steps that were taken to overcome them. Through chapter (2), (3) and (4), our efforts on modelling data analysis with Radial Basis Functions, couched as a non–linear structure, converged in a somewhat unexpected way, yielding our own contribution in a rather different way, one that offers interesting answers to problems of rules selection, clustering, model reduction and the complication introduced when the noise is correlated. Another strand of our work is to discover explanatory principles, stochastic or mathematical, often hidden and abstract, to make sense of certain phenomena that seem to neuro–fuzzy folks difficult to understand and lacking meaningful pattern. The cases in point are rules selection, how many hidden neurons should a neural network have, whether the number of training data is sufficient and which data points should be used.

Our data sets are different from the data–sets often used to illustrate new algorithms and techniques in neuro–fuzzy research. The thesis data-sets were also of interest to our industrial research partners: BT, SmithKline Beecham, Glaxo-Wellcome and most importantly IBM Super computer Research and Development organisation. Such data are very large and complicated, sometimes with thousands, sometimes with many fields, often up to or over one thousand for each entry. Such data also include complicated discrete count data from electricity companies and record of daily customers insurance claims (Davis 1993 ,Dassios and Embrechts 1989), As in insurance firms such as Lloyds/TSB (Mallison and Bentley and Maana 1999)[1] and described in (Davis 1993), claims in insurance company is the amounts of money with some random sizes, arriving at some random instants. The premium income is similar to bandwidth in queue problems in the sense that the path of the inventory follows a deterministic trajectory when no demands arrive. These processes are powerful enough to account for data obtained from simulation of queueing traffics (Bischak, Kelton, and Pollock 1993).

Similarly, the paraquat production process of Astra–Zeneca, is a complex, multi–stage, batch production process that is also expected to produce large, complicated

[1]These references are contained in Computer Science in University College London and Statistical laboratory in Cambridge University as internal reports.

daily production data-sets. Finally, in the fast moving e–Commerce goods area of interest to Unilever, the examples of food product data they have provided, are neither large nor complicated, but are small, sparse and, because of the way items are derived from human preference panel data, discrete and noisy. Such data–sets are also difficult to use and not well represented in academic research.

In each case the observed data can be represented in the form $X_t = (n_t, x_t, t)$, where n_t the number of customers that have arrived in time t and x_t is the amount of demand or total work load brought in by the customers. For the daily claim data, n_t will be the amount of customers that reported and took out claims of total size x_t at date t. In insurance data set, also given in the appendix, there is a steady inflow of premiums, a random output of claims at random times. The daily number of claimers and the total daily claim size are given.

To put in another way, suppose a periodic review insurance system for which the claims $x_1, x_2, \ldots,$ for a single insurance type in successive weeks are independent random variables having a common probability density $f(x)$ with finite moments. Similarly, the count random variables $n_1, n_2, \ldots,$ are the number of individual customers who put the requests in successive weeks. For this purpose, consider n independent renewal processes $\{N_k(t)\}, k = 1, 2, \ldots, n$. In fact, in an insurance data set, $N_k(t)$ is the number of claimers of a certain claim type k that arrive during a time period $(0, t]$.

The development of the internet at the current level of recent technological advances, telecommunications and computer networks, the superposition of demand processes is taken up by the *multiplexing* of various traffic sources over different times and space, to use the words of (Kelly 1996).

The problem of predicting the traffic work load over time have been portrayed as a "landmark in data communication systems," opening the stage to a range of schools of thought with no precedent, each claiming victory over the other. Within the queue theory and applied probability school, the concept of predicting the super–imposed workload is called *effective bandwidth,* a concept we shall return to. In a communication networks, like an ATM (Gibbens Kelly and Key 1995, Hsu and Walrand 1994), the number of calls received at the switch is a superposition of the individual subscribers, each represented by a renewal process $\{N_k\}$. Let $N(t) = \sum_{k=1}^{n} N_k(t)$ be the sum of these arrivals. Denote by $\{F_k(t)\}$ the probability distribution function of the intero–ccurrence time between consecutive arrivals of type k and denote by α_k the first moment. The waiting time of the process $\{N(t)\}$ is $\tau = \min(\tau_1, \tau_2, \ldots, \tau_n)$, where τ_k is waiting time of each process. Also, figure (1.1) illustrates this set up with

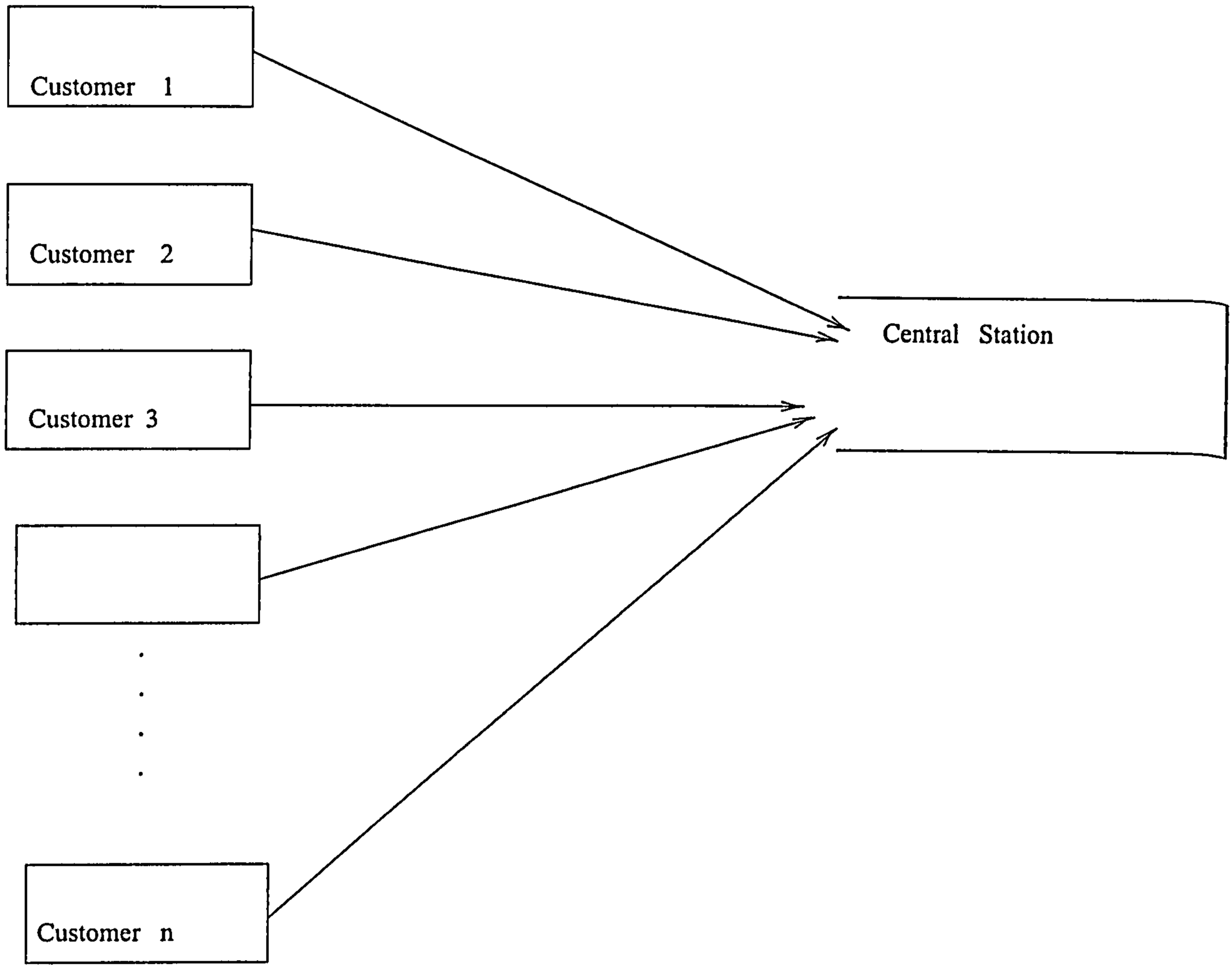


Figure 1.1: Superposition of renewal processes

n customers representing the renewal processes and the overall demand/inventory system is a combination of effects of the individual processes.

In the applications of standard Neural/RBFs regression methods to our data sets, the modelling problems encountered differ so much in character, and the criteria imposed on the noise differ enough in nature that it is not possible for us to limit ourselves entirely to the standard assumptions, usually coined as " because neural networks, fuzzy logic and RBFs are non–linear, they will be able to approximate these data well enough." We had to ask some searching questions about the nature of the processes generating these data records and question certain standard common sense assumptions in neuro–fuzzy applications.

The commonsense notion is that most data collected from industrial processes can be accurately arranged as input/output data to be used to train an appropriate neuro–fuzzy network such that it can be used to predict the future from the past (Chen

and Billings 1992, Jang 1992, Jang 1993, Berenji, Chen, Lee, Jang,and Murugesan 1990). To avoid misunderstanding, we are not speaking here of industrial data of all types, but rather data from count processes of a renewal type, and even here we will not attempt a full analysis of renewal processes but rather will give a somewhat three idealised conditions that are clearer in retrospect than they were at the time when this investigation started. We want to consider, two major shifts, one that inaugurated the applications of RBFs, and a second, rooted on renewal theory, that is now dorminant and that offers some new perspective on the traditional modelling problems we first had.

This thesis examines how to model this class of processes under the following conditions:

1. when the number n of the renewal processes and individual renewal count $N_k(t)$ are very large.

2. when the number n of the renewal processes gets very large and in each individual renewal process the renewal occurs very rarely. In this case *the combined effect is due to many small causes.*

3. when the number n of the renewal processes is small and in each individual renewal process the renewal occurs very rarely. In this case *the combined effect is small but is due to very many other small causes, most realistically occurring in discrete times.*

We noticed in the course of our investigation that neuro–fuzzy modelling could not deal with cases (2) and (3) above, because of its implicit reliance on uncertainties that were not quite Gaussian in nature as in the case (1). We first encountered case (2) and (3) in modelling BT data sets with neuro–fuzzy networks. The data sets were in the standard form $X_t = (n_t, x_t, t)$, as explained above. In course of the investigation, we advanced a discrete dynamical structure from control theory (Davis and Vinter 1984) and demonstrated how it could be used to model the process $\{X(t)\}$. We provided answers to the form in which neuro–fuzzy system could emulate this dynamical structure. This work later was refined to investigate the adequacy of RBFs and how to sharpen it for model reduction and non–linear optimisation. In part, case (1) resulted in a lot of success and the results of predicting $X(t)$ were quite impressive. Unfortunately, our neuro–fuzzy system failed very badly in cases (2) and (3). With regard to why Neuro–fuzzy approaches were failing, it was widely held that their inherent non–linearity were sufficiently endured to model whatever

complexity might be in the process, including the modelling of queue length as in (Chen and Leslie 1992, Habib 1995, Hiramatsu 1990).

In the first case, the actual counts are very large and their superposition is large. From the *central limit theorem,* this can be approximated with a Gaussian process, enabling the use of Radial Basis Functions (RBFs) as in the next few chapters. We applied theoretical discrete time series framework developed in (Davis and Vinter 1984) for model representation and reduction.

The second case provides the basic ingredients that support the theoretical explanation for the occurrence in many industrial processes of Poisson processes. For further remarks on this limit theorem see (Cox 1962, Cox and Miller 1965). Davis (Davis 1984) extended the study of these processes in very general context and developed a completely new approach in continous–time. His theoretical basis is about Markovian models in which the renewal process $\{N(t)\}$ is defined in a similar way to a Poisson process. However, inter arrival time are i.i.d random variables with some general density function (not necessarily exponential). The renewal process is close to a Poisson process. Although, it is not a Markov process it has been shown by (Cox 1962) that it can be made into a Markov process through the method of adding an additional state. Davis extended the idea and provided the means of evaluating the performance of such systems. The generator for continuous time renewal processes was derived for the first time in (Davis 1984). There it was shown that such renewal processes fall under a general class of processes known as *Piece wise deterministic Markov processes* (PDPs). The generator of this renewal process is usually an integro–differential equation, providing, at least in theory, with the method of computing expected values of various costs associated with the process.

In the third case, the process is neither a Poisson process nor a continuous process. However, we have developed a new method of modelling such discrete processes by making extension to Davis theory; and we develop theoretical basis for modelling a PDP process in a discrete setting. This new formulation allows many Markovian systems that are discrete in nature, not amenable Davis's continuous–time theory, to be modelled by this discrete time Markov chains. *This brings us to the title of this thesis: this thesis is about Markovian models, and particularly about their applications to data analysis and queues.*

A Markov model is, in our case, a collection of random variables $\{X_t : t \in (0, 1, 2, \ldots,)\}$, where X_t is a scalar or a vector of dimension 2 or 3. The essential aspect of Markovian modelling as opposed to any other modelling framework is that the random variable X_t is forgetful of all but its immediate past. It is sufficient to indicate that

the stochastic process $\{X_t\}$ is evolving in a given state space E with a transition probability matrix $P = \{p_{ij}\}$, to be a time–homogeneous Markov chain. There is a set of *transition probabilities* $\{P^n(x, A), x \in E, A \subset E\}$ for some given set A such that for times $n, m \in (0, 1, 2, \ldots,)$

$$P\{X_{n+m} \in A | X_j, j \leq m; X_m = x\} = P^n(x, A). \tag{1.1.1}$$

In fact, $P^n(x, A)$ denote the probability that a process at state x will be in the set A after n steps or transitions. The independence of P^n on the history $X_j, j \leq m$ is the Markovian property.The data analysis and queues considered in this thesis illustrate the scope of the applications of Markovian modelling.

In the data analysis case, the electricity consumption X_t at discrete time t can be a function of its past values $X_{t-1}, X_{t-2}, \ldots, X_{t-k}$, perturbed by the volatility of the customers's preferences and habits inherently affecting demands, which is incorporated as disturbance term e_t. Figure (10.1) in the appendix depicts electricity consumption over a period of about three days, sampled minute by minute. This represents a 4,500 point time series. The consumption is accounted for by industry for 50% and by individual consumers for the other half. There are more than 10 million individual customers, partitioned into n industrial/residential customers, so that at each sample time t there are $N(t)$ active customers in sector k, generating a renewal $\{N(t)\}$ at inter arrival intervals $(t-1, t]$. In this time interval amount of electricity consumed by each active customer j is $X_j(t)$ = the total electricity consumed in that time interval. Our first modelling assumption is that total active customers $N(t)$ for the electricity central board that arrive in $(t-1, t]$ is a Poisson process $\{N(t)\}$ with rate λ. The successive electricity load $X_j(t)$ from the active individual customers $j = 1, 2, \ldots, N(t)$ are positive, independent random variables having a common probability distribution $F(x)$, and are also independent of the arrival process. The generic variable x_j denotes the individual electricity load amount. Thus,

$$X(t) = \sum_{j=1}^{N(t)} x_j(t) \tag{1.1.2}$$

denotes the electricity load at time t. This is just a compound Poisson random variable.

1.2 Insurance and electricity demand processes

In this electricity data set and the insurance case considered in this work, the arrival rate λ is actually very large, making the process $X(t)$, according to limiting modelling condition mention above, Gaussian time series. We now give a justification for this.

It is often common, even though the observations are made at discrete times, to model the underlying process as a continuous–time process. Even if there is no underlying continuous process, control people still find it convenient to model the data as a continuous time process at discrete times. When they do this, they construct certain Markov processes that are defined in terms of stochastic differential equations. An instance of this is an auto–regressive process (Davis and Vinter 1984) which is understood to be a solution of a stochastic differential equation. It turns out that a first order auto–regressive process is Markov but those of order greater than one are not. However, as we shall show, higher order processes can be transformed into a Markov process. The importance of this Markovian property is the fact that predictions can be made about these processes without taking into account the entire past history of the process.

For a general demand process, consider a sequence $X_0, X_1, X_2, \ldots, X_{t-1}, X_t$, where X_s is the total demand sampled in the time interval $(t-1, t]$. The thesis prediction problem can now be stated as follows: *given the above sequence up to time t, what is the predicted demand at time* $t+1$.

Consider the stochastic process $\{X_t\}$ on R and is adapted to the natural filtration $\Im_t$ defined as

$$\Im_t = \{X_s : s \leq t\}. \tag{1.2.1}$$

This process can be regarded as a family $\{X_t\}$ that takes values in some measurable space with an appropriate Borel σ –field, according to the recipes in (Davis 1993). Then the process is a *Markov process* if for any times t, s with $t \geq s$

$$E[X_t|\Im_s] = E[X_t|X_s]. \tag{1.2.2}$$

This implies that the evolution of the system is independent of the past. Thus, the system is forgetful of all but its most immediate past.

Suppose, following ideas from control theory, the system is j order system but non–linear. Then the state at time $t+1$ depends on j lagged states before state $X(t+1)$. That is, $X(t+1)$ is some non–linear function $f(.)$ of the states $X_{t-j+1}, X_1, X_2, \ldots, X_{t-1}, X_t$,

The autoregressive model

$$X_t = f(X_{t-j+1}, X_1, X_2, \ldots, X_{t-1}, X_t) + e_t. \tag{1.2.3}$$

This is central in Neural networks and RBFs modelling (Chen and Billings 1992), which captures the essential feature of such a demand system. Although, this structure is Gaussian under the operating conditions, the process $\{X_t\}$ is not a Markov process. But, by using the method of supplementary variables, we construct a new state $Z_t = (X_{t-1}, X_{t-2}, \ldots, X_{t-j+1})$. The process $\{Z_t\}$ is now Markov. Thus, the Gaussian process in (1.2.3) becomes

$$Z_t = f(Z_{t-1}) + e_t. \tag{1.2.4}$$

We have chosen the RBFs to approximate this non–linear function. Our work on RBFs is based on trying to solve the following problems:

1. Model reduction problem: this problem is concerned with how to choose the number of basis functions that are needed to approximate the mapping function. Too few or too many hidden nodes may result in either an unsatisfactory generalization error with respect to new data or a poor fitting with respect to training data (Lyung 1995, McKay 1992).

2. Optimisation problem: the approximation problem is very hard to solve because of the optimisation complexity of selecting the centres of the Gaussian basis functions and the non–linear constraints involved in the optimisation.

3. Correlation problem: Are the noises correlated and if so, how can we still solve the model reduction problem?

4. Non–stationarity problem: Is the process stationary or transient? If the process is transient how can we predict the stationary value. We found that these processes are volatile in such a way that the stochastic properties such as the variance changes with time.

1.3 Data communication across computing switch

Clustering problem in computing involves the study of message passing protocols that enable communication among nodes using independent operating systems. Message passing allows blocks of memory data to be moved between nodes under software control. In the IBM switch SP protocol and InfiniBand (IB) protocol studied

in this thesis, software sets up the type of communication required by creating an Inter–process Communication (IPC) list in memory of hardware nodes, called a *descriptor list.* Once this list is set up, the software component in the Node hardware tells the adaptor connected to the I/O bus to do the actual data movement while it is off executing other functions. In the SP protocol implementation studied here, each node is an IBM new multi–processor system, called the Regatta system, which has 32 CPUs and can have up to 8 independent adapters connected to the node's I/O busses. Figure (1.2) shows the high level diagram of how the nodes are connected to the switch via the adaptors. Each adaptor provides one port to the switch and it is visible to the software layers above it. A CPU user code can

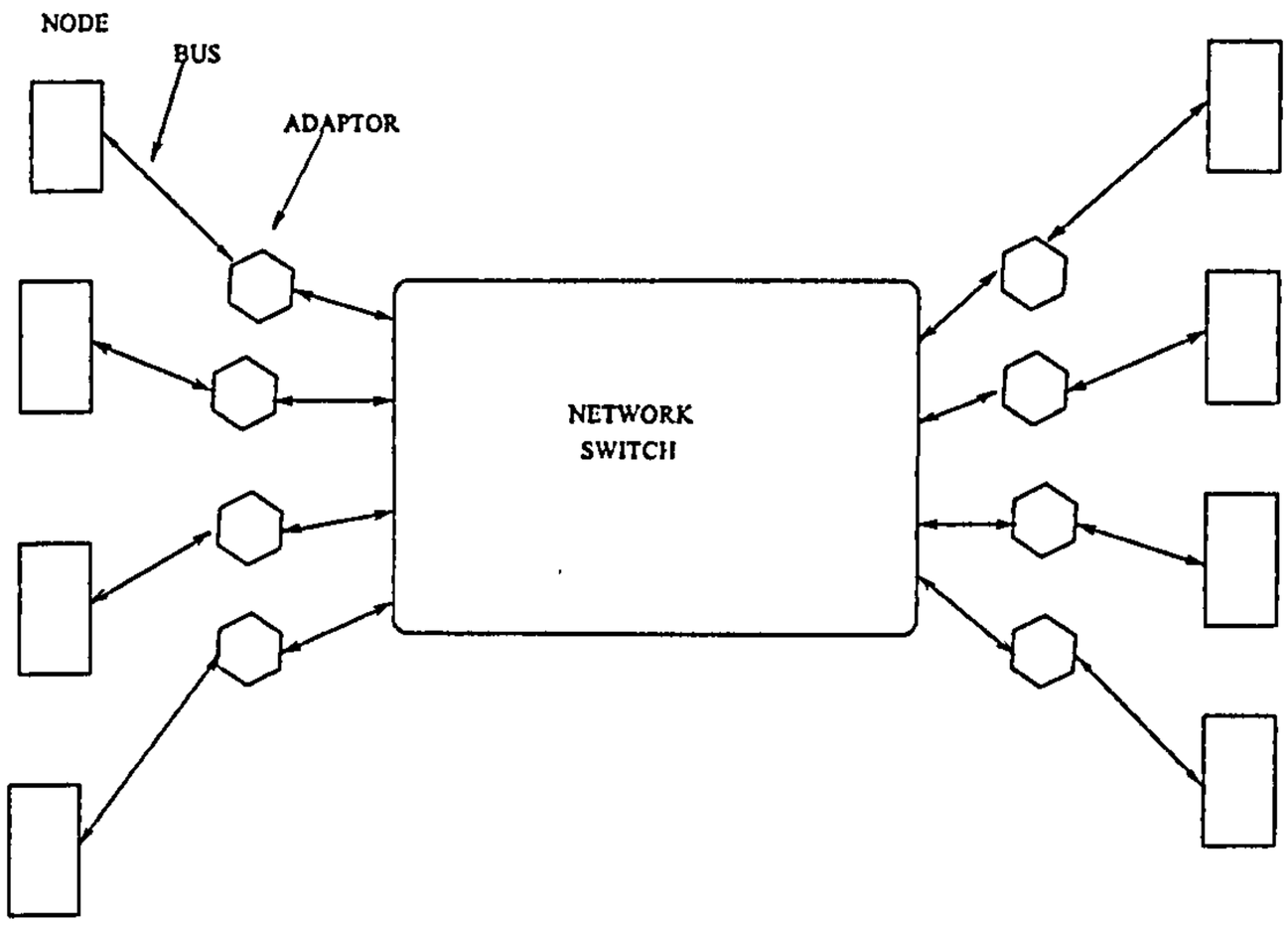


Figure 1.2: Clustering computers for message passing

use the message passing facilities after it has been given a channel by operational system. The operation system prepares the facilities associated with the channel. These facilities include the allocation of memory for descriptors and data. These must be done before the CPU issues a StartDma command. A StartDma command tells the hardware adaptor that CPU software has defined a compatible descriptors on both the sending and receiving sides. The StartDma command will supply a channel pointing to descriptors for the required functions.

Once a channel has been created, user level software can use the channel for data communications without help from the CPU. A user's perspective of a channel is that it establishes a memory region that both it and the adaptor hardware can access. It is important that the memory region containing the data is pinned which makes it possible for messages to be exchanged with tasks running in other nodes and the descriptors providing the adaptor with the type of operation necessary.

The software and the firmware can be programmed to provide a number of logical channels, each with its own descriptor list and operating from other channels. A very popular message passing protocol is the message passing interface (MPI) which is used to exchange large amounts of data used by parallel processing tasks running on different nodes.

The InfiniBand architecture is a switched–fabric architecture for a new wave of I/O systems and server systems. The InfiniBand Architecture (IBA) promises to replace bus–based I/O architectures, such as PCI, with a switched-based fabric whose benefits include higher performance. It will also provide the facility to create networks of servers and shared I/O devices. This is a product of the InfiniBand Trade Association (IBTA), an industry group of over a few hundred companies led by Intel, Compaq, Microsoft, Hewlett-Packard, Dell, IBM, and Sun Microsystems.

Like the IBM SP protocol, each node contains one or more ports that are connected to other ports with point–to–point links. Three speeds of InfiniBand bi-directional interconnection links are supported: x1 (2.5Gb/s), x4 (10Gb/s) and x12 (30Gb/s). The InfiniBand transport layer defines protocols for both unreliable and reliable messages, as well as Direct Memory Access (DMA) mechanisms. These protocols enable a variety of I/O and IPC operations between end nodes. The interface for managing message traffic within a channel adapter is called a Queue Pair (QP) and each QP contains both a Send buffer and a Receive buffer for message passing. Communication channels between nodes are established by connecting their QPs. In IB the node adaptor is known as a *host channel adaptor* (HCA) and it resides on a host processor node and complies with the transport software interface.

In addition to the SP switch, IBM is developing a number of HCAs and we have considered these as a case study in the application of queues in chapter(8).

1.3.1 Client and server clustering

There is a proliferation in the use of client and server models in data communication among computing nodes. The system architecture is large, but the essential structure is that customer applications are split into local tasks, each running on a "client workstations, allowing resource–intensive computations to be run on "server SMP machine. The user demands interact with the host server through an adaptor connected to a switch. This Switch sends that requested information to the host server. In the last few years, SMP machines have been replaced as a server in distributed clustering. The original PC networks were based on file sharing architectures, where

the server loads files from shared location to the desktop environment. The user's job then runs in the desktop environment. In the 1990s, PC LAN computing changed because the capacity of sharing file was limited as the number of online users grew. Now such method can only support about 12 users. PCs are now being used in a clustering architecture.

Suppose our n client demands are independent renewal processes, each with its own distribution and mean α_k; the superposition theorem asserts that, when the means of the individual processes tend to infinity in such a way that sum

$$\frac{1}{\alpha_1} + \frac{1}{\alpha_2} + \ldots \frac{1}{\alpha_n} = \lambda$$

remains constant, when $n \to \infty$, we argued in chapter (2) that the resulting process tends to have the exponential distribution with mean $1/\lambda$. This supports the claim that the superposition process is Poisson. This shows that the demand process is a Poisson process under the stated conditions.

1.4 Queueing model of data communication

In the IBM MPI protocol under development, messages are transmitted from the sender Node to the receiver Node at a constant rate of $u > 0$ whenever the inventory of the untransmitted packets in the protocol *window* is positive. The concept of window is similar to the flow control window used in TCP/IP protocol, where at any instance in time, the sender maintains *sequence numbers* corresponding to packets that are in the pipeline of the link and have not been transferred and acknowledged. It will be realistic for the bandwidth u of the link and the average message size $E[X]$ to satisfy the stability condition $u > \lambda E[X]$, that is the transmission rate through the I/O bus and the adaptor rate is larger than the average amount of packets offer to the protocol window per unit time. Thus, we are interested in the time dependent inventory of untransmitted packets in the protocol Window.

In chapter(8) we shall show how this system can be modelled as an $M/G/N$ queue. In addition to the uncertainty inherent in trying to predict the queue length of packets in the pipe line channel of communication, we are faced with the problems of estimating the costs of processing , opening and sometimes closing of channels of communication. These are all new territory for queueing theory applications.

There has been a recent paper (Davis and Howl 1997) in which the application of

PDP theory to queues have been demonstrated using the $M/D/1/N$. Davis and Howl studies the generator equation associated with $M/D/1/N$ and set out a procedure for computing the transient queue length and its distribution. They were motivated by the fact that $M/D/1/N$ are an important component of modern communication system. They envisaged that communication of data messages across asynchronous transfer mode (ATM) occur in some fixed–length blocks called "cells" over buffer communication channel. Thus, the performance of such systems require the use of $M/D/1/N$ queue model. The importance of our extension of this idea to more general queues such as $M/G/1/N$ is that we can tune the speed u of our model to be in line with the link bandwidth of the channel and extend the modelling framework to a wide range of message passing applications on different range of protocols. The further advantage of this Markovian framework is that it provides a context for messages of various distributions and for analysis of communication protocol of different packet sizes.

Davis devoted his effort to the study to the control theory for such processes (Davis 1993, Davis, Deemster, Sethi and Verges 1987, and Davis 1991) rather than on the derivation of performance results of these processes. Similar line of work were carried out under the name " *control theory for PDPs* (Dempster and Ye 992, Costas 1993).

Here, again as in the past, too much attention were given to the broader questions about the existence of optimal control of PDPs and very little to the practical solution of specific problems. The method of analysis were similar in spirit to the problem formulation, the search for necessary and sufficient conditions for optimality, optimal stopping rules and other formulation and results reported in (Davis 1993).

The scope of queueing problems circumscribed by the results from $M/D/1/N$ is much narrower and so in this thesis we have extended recent development to the $M/G/1/N$ queue. This extension is based partly on a truncation argument that the failure rate is not constant, but it depends on the elapsed time since the last failure. This service is completed either upon failure or when the elapsed time hit a fixed boundary point. The $M/G/1/N$ queue is more general than $M/D/1/N$ and the results are more applicable to a much wider class of queueing problems, including methods for tuning and parameterisation of queues with arbitrary distributions. . The analysis incorporated nearly all single server queues with general distribution. The theory can also be extended to two servers $M/G/2/N$ queue and the numerical algorithms for solving the resulting system of partial differential equations have been developed. Predating the interest for $G/G/1/N$, the algorithm for solving the two

servers $M/G/2/N$ queue has been modified for the $G/G/1/N$ queue. This is a first significant step towards producing general transient models for queues. We used the PDP theory to derive a system of partial hyperbolic differential equations and introduced numerical methods for solving them.

In the field of queueing theory (Asmussen 1987) some efficient impressive methods have been developed for calculating the stationary values of parameters of interest, making particular use of special mathematical structures. With transient problems, the situation has been rather different, particularly, if one wishes to study the expected queue length at a given time and interested in the effects of the actual speed of the server on the queue length. There are no unified model dealing with this problem. Thus, the transient $M/G/1/N$ queue problem introduced here is an instance, which falls outside the scope of any currently available model in the study of queue. The body of knowledge developed for $M/G/1/N$ provides a framework within which many other queueing systems can be studied.

In many applications such as $M/G/1/N$ a series of studies (Davis 1984, Cox and Isham 1986 , Dassios and Embrechts 1989) have identified the set of integro–differential equations for modelling queue lengths, making essential use of the special structure of the so–called *virtual waiting time* (VWT). In the VWT model, a job arrives with Poisson rate to meet some unprocessed load that has to be done before it enters service. Cox and Isham (Cox and Isham 1986) began the first analysis of $M/G/1/N$ which employed the application of PDP theory developed for VWT processes (Dassios 1989). This work gives further insights into the modelling values of PDP, but it had similar consequences: the resulting functional equations were hard to solve in the transient case, relegating the analysis to steady state case. There are simple ways to evaluate the fact that the VWT fails to address a wider problem scope.

It has been pointed out that the sort of integro–differential equations (IDEs) that arise when studying the generators of $M/G/1/N$ using the structure of the (VWT) are poorly understood. The VWT methods of PDP theory cannot be unified into body of tools to be applied to queueing problems. The difficulty in dealing with the IDEs in the theory of PDP is not new. In fact, it was recognized long ago (Kelly 1984) that it is the lack of understanding of the IDEs arising from renewal theory (Cox 1955,) and Piece-wise linear (PL) theory (Gnedenkho and Kovelenkho 1966) that prevented a large scale applications[2] of PDP theory. Gnedenkho's process is similar to PDP except that the deterministic process is less general. They used this

[2]PDP theory unified renewal and PL theories.

process to study problems arising in queues. Their queue model gives extended a formulation of the (VWT). This gives an integro–differential equation for the waiting time distribution for Poisson input and general service time in the transient state. The general results for the steady state were obtained, but it was difficult to compute the transient case.

The major departure made in this thesis on the $M/G/1/N$ from its predecessors is that it is entirely based on the application of PDP to renewal processes, providing a general method of converting a non–Markovian queueing process. This current development based on the assumption that a large class of queueing problem, including the $M/G/1/N$ test example, can be reduced to the investigation of a system of partial differential equations (PDEs) of which there is a better knowledge than IDEs. In addition, the resulting PDEs in $M/G/1/N$ queue is general to the extent that it provides both stationary and transient solutions within the same framework. In addition, there is a *duality* relationship between transient distributions of queue length and their actual expected values. The solution to any of these performance measures rests on judiciously tuning parameters in the resulting PDEs.

There is no doubt that Renewal theory provides the unifying tools for studying many stochastic processes. This possibly explains why several research monographs and books introduce renewal theory first: Heyman and Sobel (1982), Kohlas (1982), Ross (1983), and (Wolff 1989).

The study of Martingale calculus has found its greatest successes in control, first for diffusions, then for jump processes. It has also found its way into some of the classical areas of queueing processes, an account of this is given in the book by (Bremaud 1981). There is no doubt that PDP processes go a step further in its attempt to incorporate all non–diffusion models of applied probability in one framework.

Despite the riches of the PDP theory in providing an excellent framework for modelling most systems, it is now generally accepted that the amount of genuine applications of the theory is disappointingly small. It is equally well admitted that the general problem centres around the sort of integro–differential equations that arise when studying the generators of these processes. The generators of PDPs are poorly understood. There seem not to be a unifying framework that links these generators. Each generator is specific to the application domain upon which it is derived and a slight change in boundary conditions leads to a completely new generator.

1.4.1 Our strategy for $M/G/1/N$ queue

The first rigorous developments of this modelling framework used the $M/G/1/N$ as an illustrative example. In an $M/G/1/N$ queue, calls arrive as a Poisson process with rate λ. The service time $\{S_n\}$ has a general distribution G, and G is continuous and has a hazard rate function $\mu(t)$. The buffer has capacity $N-1$. This is a single server queueing system, where calls arrive from infinite sources into a single node with one channel and mean holding time $(1/\mu)$. The first call that arrives into an empty system goes into service immediately. Let T_n be the inter–arrival time of the customers. Hence, T_n is the time between the arrival of customer number $n-1$ and customer number n. Let x_t be the number of jobs in the system at time t. Because the process $\{A_t\}$ is Poisson, its inter–arrival times $\{T_n\}$ are independent and exponentially distributed. The "memoryless " property of the Poisson process ensures that $\{A_t\}$ is a Markov process. Hence, this explains the attachment of M in the queue model $M/G/1$. However, the process $\{x_t\}$ is not a Markov process.

1.4.2 A new discrete PDP theory

Davis's treatment of PDP Markov processes is in such a high plane of sophistication, and it is in such a general context of control theory that it is hard for those interested in applications to see what is going on. Most of the studies of stochastic control theory rely on Brownian motion in particular and diffussion processes in general. Considerable progress has been made in the study of these processes, partly because most problems can be reduced to the study of partial differential equations which are well understood. Brownian motion is mainly concerned with the probability laws governing the time and position of a Markov process when it first visits a subset of its state space. These probabilities satisfy partial differential equations that are entirely well analysed in the literature. Whereas, for PDP Markov processes the study of queue length probabilities, particular in transient cases, involve delicate integro–differential equations that are hard to analyze.

This work will show that these difficulties evaporate in the case of a discrete theory. As we shall show in the presentation of the new theory, for arbitrary subset of state space of a lattice points, the time and queue length or renewal position are automatically measurable, and the integro–differential equation encountered in Davis's continuous theory reduces to partial difference equations of a *random walk fashion.* In this sense the study of the PDP Markov process leads one to partial difference equations in a very simple setting.

We have developed this theory in chapter (7).

1.5 Neuro–fuzzy applications to data analysis

In the 90's came a renewed interest in non–linear system identification under the framework of neural networks and fuzzy logic , a field now known as neuro–fuzzy techniques. This trend is based on the application of neural networks, fuzzy logic and non–linear optimization. A fact that contributed to the first upsurge of the applications of fuzzy logic is the simplicity of the fuzzy algorithms developed in (Mamdani 1976). As far as Neuro–fuzzy techniques is concerned recent developments can be found in (Brown and Harris 1994, Jang 1992, Wang 1992). Despite the resurgence of interest in the applications of neural networks and neuro-fuzzy to the learning of dynamical systems, there is no systematical way of ensuring the selection terms to be included in the model.

A characteristic feature of demand and inventory system is that their behaviour changes with operational conditions and the input–output data of the system is often changing. This introduces a number of problems for the design of learning algorithms. Design of Gaussian neuro–fuzzy networks requires the adjustment of centres of membership functions (Linkens 1993). For data analysis applications, this requirement devolves to the selections of the centers which sufficiently sampled input data. While reasonable learning can be achieved by appropriately sampling linear system (Slotine and Li 1991, Sanner Slotine 1992), certain difficulties are introduced for model selections. The number of hidden nodes is very important for determining the training and generalization errors performances of these artifacts. Recent work suggests some formulation that makes us understand this problem intelligibly, but not much more. Fuzzy logic provides another solution to this problem by imposing constraints on certain rules. Unless these constraints are imposed , the complexity of the search for an optimal learning structure will explode.

1.5.1 Model reduction

In data analysis using RBFs, building or extracting a model that is sufficiently accurate and reliable is difficult. A 'brute-force' approach in which, all combinations are tested to find the best features for describing the data, cannot be used on the large data sets because of computational complexity. This approach fails on the sparse and highly noisy, correlated and *heteroscedastic* data sets because of

inadequate information.

This thesis contains a Markovian time series model of this problem. It is based on a process model sufficiently general to encompass the application scenarios of interest to all our major industrial partners and is here described in terms of a machine learning approach to enable us to illustrate the main ideas within a single framework.

It is of importance to note that our Markovian scheme for discrete time series may also arise when a physical quantity does not have an instantaneous time value but has meaning only when *accumulated* or integrated over a suitable time interval. In fact, the extension of Markovian systems to image processing, where time is substituted with pixel values, is well noted. Other examples of such Markov random field, where time is substituted are the daily insurance claims, yields of consecutive batches in an industrial process like in the Astra-Zeneca or the values of preference scores over different tea types in the Unilever data set.

The gap that exist between such models and the real systems is often called model uncertainty. This uncertainty is different from exogenous actions from noise. The second cause of uncertainty is the alteration of system behaviour due to changes in the noise distribution. The motivation for Radial Basis Functions is derived from the determination to develop models which track system non–linearity. In addition, RBF is simpler in the sense that it is non–linear in its structure but linear in regression parameters, making it amenable to the application of model reduction principle introduced in (Davis and Vinter 1984).

Davis (Davis and Vinter 1984) were among the first control theorists to adopt stochastic modelling towards the solution of model reduction problem in general regression. Their theoretical principle, based on stochastic control theory, was to apply hypothesis testing using F–distribution to the selection of terms in linear regression problems. They formulated criterion for selection terms to be included in a dynamical systems that can be modelled as a system of discrete equations. This theoretical approach, in conjunction with other studies (Segall and Kailath 1973) in system identification within the framework of control, began to provide a firm foundation for the investigation of observed patterns in data.

We wish to consider a large class of uncertain systems with r dimensional inputs u,

and m dimensional outputs y, defined by the column vectors:

$$y(t) = [y_1(t), \ldots, y_m(t)]^T, \tag{1.5.1}$$
$$u(t) = [u_1(t), \ldots, u_r(t)]^T, \tag{1.5.2}$$

each labelled by a discrete index $t(= 1, 2 \ldots)$. Such systems may be modelled as a discrete non-linear regression according to an equation of the form (Zhu and Billings 1996):

$$y(t) = f(y^{t-1}, u^{t-1}, \epsilon^{t-1}) + \epsilon(t) \tag{1.5.3}$$

where,

$$\epsilon(t) = [\epsilon_1(t), \ldots, \epsilon_m(t)]^T \tag{1.5.4}$$

is the noise on the outputs, $y(t)$. And y^{t-1}, u^{t-1} and ϵ^{t-1} respectively denotes output, input and noise sequences as follows:

$$y^{t-1} = [y(t-1), \ldots, y(t-n_y)], \tag{1.5.5}$$
$$u^{t-1} = [u(t-1), \ldots, u(t-n_u)], \tag{1.5.6}$$
$$\epsilon^{t-1} = [\epsilon(t-1), \ldots, \epsilon(t-n_\epsilon)]. \tag{1.5.7}$$

In general, $f(.)$ is a non-linear, m–dimensional vector function of many variables, so the input–output relationship is a non-linear mapping which may be very complicated and, in practice, is often poorly understood.

The index t may be thought of as a discrete time index, but it does not necessarily have to represent a time as, for example, in the tea structure–activity data of interest to Uniliver[3], and may, in principle, be any index to the discrete data entries. The form adopted in equation (1.5.3) is very general and allows the possibility of complicated, non-linear relationships between $y(t)$ and the sets of inputs u^{t-1} and outputs y^{t-1}. In pattern recognition, a more homogeneous functional relationship of the form:

$$F(y^t, u^t, \epsilon^t) = 0 \tag{1.5.8}$$

may be preferable, but for process data where t is a time index, it is convenient to specialise the output at time t when (1.5.8) allows us to include the dependence of the output on the temporal history. In such cases, equation (1.5.3) represents a

[3]See Appendix.

non-linear, auto-regressive, moving average model with exogenous (external) inputs of the NARMAX kind introduced by Billings, see for example, (Chen and Billings 1990, 1992) whose notation we have followed. In the process modelling and control literature, it is usual to restrict the inputs u appearing in the function on the right strictly to *previous* inputs u^{t-1} in order to enforce causality, although in signal processing formalisms $u(t)$ may frequently be included in the inputs. In equations (1.5.6 — 1.5.7), when t is a time index[4]:

- n_y is the number of lags in the output,
- n_u is the number of lags in the input,
- n_e the number of lags in the noise, normally of course the same as n_y.

In applications in process modelling and control, it is often sufficient to use a second–order system with $n_y = 2$.

The other specialisation in equation (1.5.3) is that, as in treatments of simple linear regression, it is assumed that the inputs u are noise-free, so that $\epsilon(t)$ has dimension m, the same as y. Again, it would be easy, for example in the kind of formalism indicated in equation (1.5.8) above, to extend the notation to allow also noisy inputs by defining $\epsilon(t)$ to be a vector of length $m + r$. However, this would unnecessarily complicate the analysis at this stage: so, for the present we remain with $\epsilon(t)$ as defined in (1.5.4).

In addition, we shall assume here in (1.5.3) that the noise $\{\epsilon(.)\}$ is independently distributed as a Gaussian with zero mean and variance ρ, although the formalism could be extended to incorporate more complicated models with correlated, coloured noise processes. However, even when $\{\epsilon(.)\}$ is Gaussian, a further complication in equation (1.5.3) is that it contains a noise term ϵ^{t-1} that correlates with the output sequence y^t. Since, in general the form of $f(.)$ is in any case not known, it is usual (but see for example the work of (Seber and Wild 1989) for an alternative approach) to seek an approximate regressor in the form:

$$y(t) = \hat{f}(x(t), \theta) + \epsilon(t). \tag{1.5.9}$$

[4]In the pattern recognition case, when t does not necessarily label time, the dependence of y on u ,although not causal, may still only involve small subsets of the variables, as for example, in a Markov random field process. There is thus still a reason to retain the nomenclature introduced in (1.5.6) — (1.5.7) as in equation (1.5.8).

where $\hat{f}$ is an approximate non-linear mapping that does not include the noise ϵ^{t-1}. In equation (1.5.9), $\hat{f}$ depends on a set of parameters represented by the vector:

$$\theta = [\theta_1,\ \theta_2,\ \theta_3,\ \ldots,\ \theta_p]^T \tag{1.5.10}$$

whilst,

$$x(t) = \left[y(t-1)^T, \ldots, y(t-n_y)^T, u(t-1)^T, \ldots, u(t-n_u)^T\right]^T \tag{1.5.11}$$

is a very long column vector constructed by writing out the required, delayed components of y contained in y^{t-1} in order, followed by those of u^{t-1}. The resulting vector x has dimension $n = mn_y + rn_u$. In general therefore, we have an approximate non-linear mapping $\hat{f} : R^n \rightarrow R^m$.

In spite of the specialisations and approximations we have introduced, the scope of this model is very large and includes, in particular:

- the linear regression model,
- radial basis function networks, and
- certain fuzzy logic/neural network systems.

The relationship to linear regression, radial basis functions, and to neural networks was noted by (Chen and Billings 1990), whilst the extension to fuzzy logic (Jang 1992, Wang and Mendel 1992) is less familiar and will be explained a little more fully below in one case that is particularly straightforward.

1.5.2 Relationship to neural networks

Since radial basis functions (RBFs) are known to be sufficient for the general problem of approximating and interpolating multi-dimensional mappings such as $\hat{f}$ (Powell 1985, Poggio 1990a, 1990b, Kim and Kosko 1992), we shall use them to illustrate our approach in the following sections. In addition, for simplicity, we shall also assume a scalar output $y(t)$ so that $m = 1$ and the long vector $x(t)$ in equation has dimension $n = n_y + rn_u$.

An RBF expansion with n inputs and a scalar output implements a mapping $\hat{f} : R^n \rightarrow R$ according to:

$$y = \hat{f}(x) = \lambda_0 + \sum_{j=1}^{n_c} \lambda_j \phi(||x - c_j||) \tag{1.5.12}$$

where $x \in R^n$, $\phi(.)$ are functions from R^+ to R of width or scale, $\beta \in R^+$, $||.||$ denotes the Euclidean norm, λ_j are the weights or parameters, $c_j \in R^{n_c}$ are the RBF centres, and n_c denotes the number of centres or basis functions. Also note that the long vector $x(t)$ in equation (1.5.11) include, if necessary "delayed" values of both the outputs y^{t-1} and the inputs u^{t-1}.

As is well known, although radial basis functions were originally designed as a mathematical technique for function interpolation and approximation, there is a strong connection with neural networks and the RBF expansion in equation (1.5.12) may be regarded as a layered network structure as depicted in figure (1.3). If the centres are fixed, then this is just a single layer network with some non-linear, input (transducer) functions, a set of linear weights λ_j and a bias λ_0 acting on the output $y(t)$. This form is particularly convenient for statistical analysis. However, if the centres are regarded as adjustable parameters, the structure becomes a two–layered neural network and each radial basis function becomes a non-linear, hidden neuron or unit. In this case, the input (transducer) units are linear and, as usual, serve only to distribute the inputs $x(t)$ to the hidden units.

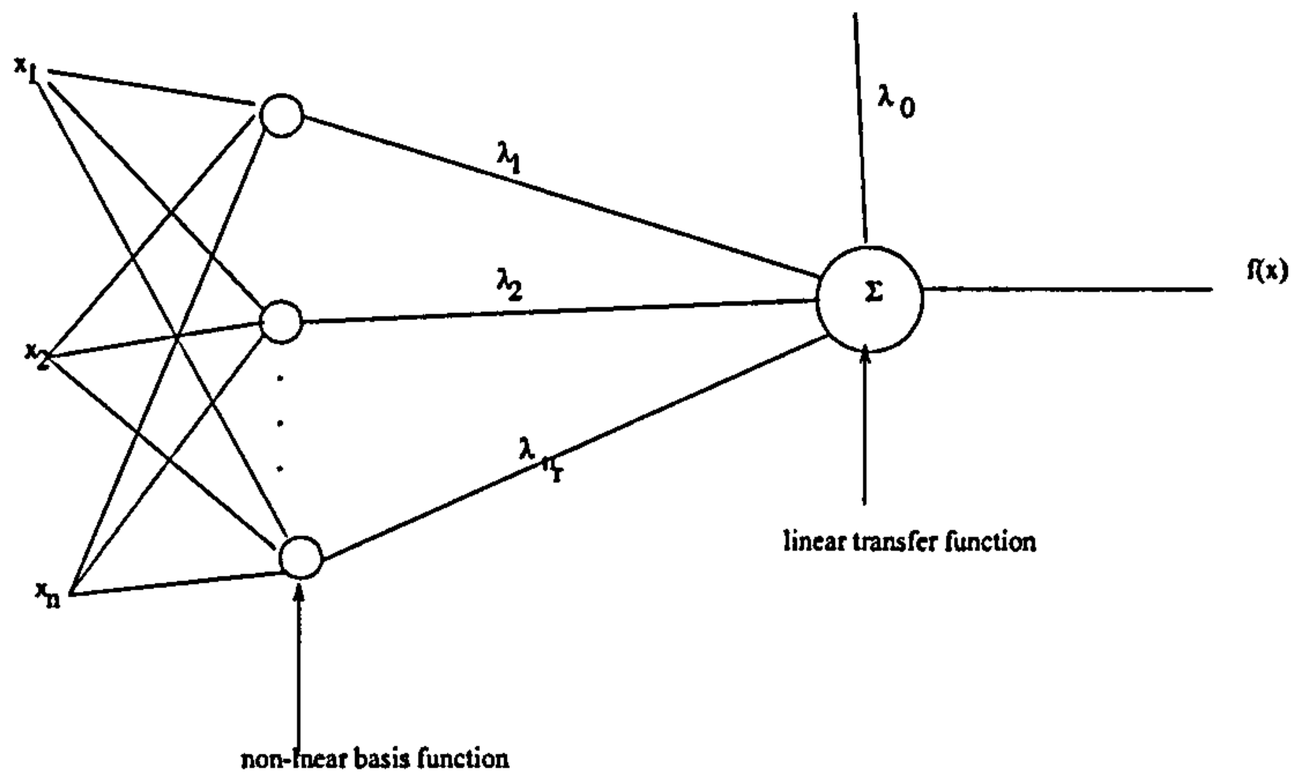


Figure 1.3: Radial basis function

Finally, it should be noted that although the network depicted in figure (1.3) is a feed-forward network(because the long vector x includes both the (delayed) sequence of outputs y^{t-1} and the input sequence u^{t-1}) the structure contains cross-terms involving y and u in each basis function. In the pattern recognition case, if the sequence of outputs y^t were to include $y(t)$ and similarly, the inputs u^t were incorporated in the long vector $x(t)$, we would have a recurrent network containing feed-back terms.

1.5.3 Relationship to fuzzy logic

The model described above also enables us to make contact with approaches based on fuzzy logic (FL) which have recently found a number of applications in the modelling and control of industrial processes and household appliances. Fuzzy systems are non–linear and are therefore well suited to the modelling of complex and ill–defined systems. They also have additional advantages beyond those provided by neural networks in that they are capable of modelling approximate reasoning, qualitative aspects of human knowledge. Fuzzy–logic does this through the use of if–then rules with fuzzy membership functions in their predicates, known more simply as fuzzy rules (Mamdani 1976). Although rule based systems have a long history of application in artificial intelligent research (Zadeh 1973 1995 1978) what had been missing until the arrival of neural networks was the machinery for dealing effectively with fuzzy–rule induction from data, that would, for example, produce simple rules intelligible to a human user. Such systems are of great importance, for example, in finance and commerce where the results obtained from automatic systems often have to be subject to final scrutiny by human authorities or managers and it is essential that simple explanations in natural, human terms can be generated. An effective method for rule induction has been developed along the line of neural networks in (Jang 1992,1993, Harris 1994), and (Wang and Mendel 1992). Hybrid systems of this sort have been successfully applied in the finance and retail industries as described in (Goonatilake 1995a,b).

The simplest way of seeing that the model may also encompass such neuro-fuzzy systems, is to note that when multidimensional Gaussian basis functions are used, each radial basis function may be regarded as a product of one-dimensional Gaussians. Each of these one-dimensional Gaussians may itself be regarded as a fuzzy membership function and their product as representing the AND connective Mamdani product operator in fuzzy logic (Assilian and Mamdani 1976). h With this simplification the Gaussian fuzzy membership functions are equivalent to a radial basis function in the sense of (Wang and Mendel1992, Linkens and Nie 1992). Other choices of the AND connectivity operator, such as the MIN connective, may be used, but can lead to non-differentiability of the system's output with respect to the parameters. This can cause difficulties in the derivation of learning rules which usually rely on differentiation of the system's output to develop appropriate iterative optimisation algorithms (Yager and Zadeh 1994). A trend that is growing in popularity is *neuro–fuzzy systems* . Neuro–fuzzy network is a combination of fuzzy logic and neural networks. One architecture with the highest popularity is composed of multilayer feedforward network of nodes. Each node performs a particular function on

incoming signals using set of parameters specific to that node. Within fuzzy logic, one node function may be membership function computation. With ideas from the field of neural networks, it has been found that many functional mapping in fuzzy logic can be successfully approximated from knowledge of set of input–output examples. This makes it possible for rules to be learned from observations. Thus, extending the scope and flexibility of fuzzy logic. This has led to a considerable number of neuro–fuzzy applications and systems which use a combination of fuzzy logic and neural networks (Yager 1994).

1.6 Our contribution

Suprisingly, however, despite the current sophistication of control theory, the insight gained from theoretical work have, in general, had little impact on practical approaches in the machine learning and data analysis. In part, this is a consequence of the abstractly mathematical nature of the theory, which has tended to be become rather detached from its practical base. In the case of queues, Kelly (kelly 1984) noted that it was the lack of understanding of the functional integro–differential equations arising from the theory that prevented substantial applications. It hardly needs stressing that, if the theoretical work is to play a role in the solution of practical problems in demand, inventory and queueing problems, much greater emphasis must be placed on both data–oriented study and numerical study of the resulting functional equations. And that is what we set out to do in this thesis. The following is a concise summary of the contribution provided to the state–of–art in Markovian modelling and it applications to data analysis: and queues.

1. In attempting to predict the level of demand, it provides the first model reduction and optimisation algorithms that is based in Fisher F–hypothesis theory. Specifically, conditions for the demand processes being Gaussian are given. This model is used to model and make predictions in the design of message passing communication protocol. Using the properties of Erlang in confluence with the generators of $M/G/1/N$, this chapter may also be read as a first step to a little known from unimportant strain of difference–differential equations and its significance to the solutions of the problems listed in (Davis 1984).

2. Algorithms and their implementations are presented, that ensure, for instance, that the selection of terms in RBFs and models of message passing protocol can be guided by statistical principles rather than choosing terms arbitrarily. The

method of analysis of variance is implemented to solve the model reduction problem.

3. It presents a new, very efficient and robust optimisation algorithms based on orthogonal projections, including algorithms for solving model reduction problems as quadratic constraint optimisation problems. Our contribution fall under a class of optimisation algorithms studied and reported in (Tankeh, Azvine and Mamdani 1997a, Tankeh and Mamdani 1997d).

4. It develops new methods of assessing the relative merits between two non–linear Markov models.

5. It presents new ways of doing data analysis with RBFs when the noise is correlated and the statistic properties are not stationary.

6. It develops a completely new methods of estimating stationary values from transient Markov models of demand processes.

7. Methods are provided for simulating queueing and message passing system, including algorithms for evaluating simulated data.

8. It is shown that an $M/G/1/N$ system which has caused a lot of problems for modellers can be modelled as a PDP Markov process, even though the service process is not a Markov process. A transient model has been developed and it turns out that the expected queue length as a function of time can be obtained from the solution of a set of partial differential equations.

9. Numerical solution of these systems equations are provided and a computational algorithm is provided.

10. The solution of $G/G/1/N$ and the two–servers queue problem is derived for the first time. Having isolated $M/G/1/N$ queue process as a general PDP model and identified its generator, the numerical solution of the resulting partial differential equations and expectation of transient queue length for various configurations are given.

11. A discrete PDP model is developed and used in the construction of a discrete renewal process with its generator. Another reason for doing this is because the discrete principles of Markov chains provide a better vantage point from which the technical details of PDPs can be viewed, unified and compared with other stochastic models. This is the major contribution on PDP research in general and queueing systems in particular.

12. We applied this theory to a case study in data communication using MPI, parallel file systems for remote disk access and zero copy communication. The results have been helpful in I/O and communication protocol design. These results are reported in the case study in chapter(8). This is a **major achievement** that might render IBM to break the clustering World record in I/O bandwidth.

1.7 Notations used in the thesis

The following notations are used in the thesis:

- θ_i is a parameter is the Radial Basis Function (RBF) network.
- **RBF** is a Radial Basis Function network.
- y_k is the output response at discrete time k in RBF modelling.
- σ^2 is the variance of the noise in RBF network.
- R^2 is the amount of the response variability explained by the model.
- F is the overall statistics for the model.
- P is the associated significance probability.
- **AIC** is the Akaike information criteria for model reduction.
- **BIC** is the Bayesian information criteria for model reduction
- **SDP** is Semi–definite programming.
- **LMI** is Linear Matrix inequality.
- $M/G/1/N$ is the Kendal queue notation. M means that the arrival process is Poisson, G stands for a genearal distribution for the service, 1 means that there is just a single server and N is the buffer size.
- $V_k(\tau, t)$ is the value function, sometimes standing for the queue length at time t when the among of service rendered to the current job is τ and there is currently k jobs in the system. $f_k(\tau, t)$ is the running cost at state (k, τ, t).
- λ is the Poisson rate.
- k is the number of customers(jobs) in the queue. We also used the same notation as a parameter in the Erlang model that shapes the distribution function. In the later case when k increases the mean holding time also increases.

- μ_0 is Erlang service parameter with mean holding time (k/μ_0). Erlang is an example of a distribution that is not exponential.
- u is speed of the server; e.g. a bandwidth of a communication link.
- D is the truncation threshold for the service time, where the service is forced to complete.
- ϕ_0 is a control theory parameter.
- N is the buffer size in the queue model.
- $N2$ is the lower boundary in a buffering system; is a shortage level in a demand and supply system or Davis's capacity expansion problem.
- T is the terminal time.
- M is the number of step size in the finite difference scheme developed.
- *pclocks* is the processor clock ticks that have elapsed.
- **PDP** is a Markov process called *Piecewise deterministic Process,* a theory developed by Davis for stochastic modelling and control of non–diffussion processes.
- **ACK** is an acknowledgement in a software communication protocol.
- **DMA** is a Direct Memory Access type of data communication.
- **I/O** is the input and output port of a computing node. In the current IBM Power 4 server GX is an I/O bus that connects the adaptor to a switch.
- **SCI** is ring–based shared memory IEEE P1296 bus and SCI stands for Standard Coherent Interface(SCI).

Chapter 2

Model reduction and optimisation

2.1 Introduction

Our strategy in this part of the thesis is to show how we have developed methods for controlling model complexity by borrowing techniques and results from statistics, optimisation and non–linear mathematics. Most attention is paid on model reduction. This has the effect of controlling the complexity arising from optimisation of the resulting non–linear systems. In other words, our focus is on problems associated with model selection and data reduction. A lot is now known about how to estimate parameters in a given model structure, but in practice, the structure of the model is seldom known *a priori*. The problem of controlling model complexity is, in general very difficult. We use *data reduction* and *model reduction* to address many open problems concerning questions such as which terms should be included in expansion (1.5.12) of the RBFs.

Although the theory of model reduction we are going to present has been introduced in (Davis and Vinter 1984), this idea has not been extended to the field of neural networks and RBFs.

It is understood that given a known RBF neural network or fuzzy basis function structure, we can use orthogonal least–squares algorithms (Chen, Cowan and Grant 1991) to obtain the significant terms in the model structure. Our work extends to this idea.

In one of our earlier work on learning using neuro–fuzzy systems (Tankeh and Mamdani 1997b), we showed how the conjugate gradient algorithm was much efficient in certain class of problems than steepest descent algorithms and other type of al-

gorithms based on search directions. We showed how the choice of each algorithm depends on the speed of convergence, the order of convergence and the computational complexity of the algorithms and the degree of non–linearity of the objective function. We made extensive study of non–linear optimisation and their applications to non–linear neuro–fuzzy systems. We have not attempted to give a detailed account of the earlier work here. However, we have outlined the major results of our contribution, particularly on the adoption of conjugate gradient. We have presented a conjugate gradient algorithm (CG) here as an example of a search direction algorithm that is relatively very simple and perform very well on some non–convex objective functions.

In our first attempt, using data sets from BT Labs, we studied and developed systematic principles that gave the relative merits of a large class of search direction algorithms when applied to general cost functions in neuro–fuzzy systems. This classification task was important for telecommunication system development but was difficult because many of these optimisation algorithms revealed arbitrary reasons why they behaved in one way or the other.

We found that the cost function in nonlinear learning system, especially the first generation neural networks and neuro–fuzzy networks were very complex because the nonlinear membership functions introduced many local minima in the optimisation problem–a difficulty equally observed by a number of researchers(Narendra and Parthasarathy 1990, Bezdek 1981,Berenji, Chen, Lee, Jang and Murugesan 1990). As most of these algorithms were based on gradient descent, it became possible for the solution path to become trap in one of these local minima without reaching the global minimum. In addition, the convergence of these algorithms depended on the initial positions and the optimisation step length or what is called *learning rate* in machine learning. Thus, tuning the appropriate conditions for a general nonlinear cost function becomes a challenge.

Our approach is similar in its goals to those reported by (Chen, Cowan and Grant 1991) and (Nie and Linkens 1993) in the development of learning control systems using Fuzzified Self–Organising Radial Basis Function Network. However, our methods are in marked contrast in major respects.

The model reduction problem formulation assumes that the efficiency of the system is: (1) sensitive to the number of neurons or rules in the hidden layer of the neuro–fuzzy networks, (2) a function of the algorithm employed, (3) the structure of the neuro–fuzzy networks and (4) the nature of data used for training the networks. With respect to the number of hidden rules, this intoduces a dilemma. Very few

hidden neurons can lead to a problem known as underfitting and too many lead to overfitting(Sjoberg and Ljung 1995).

Although the optimisation problem in general is not necessarily quadratic, we have formulated it as a quadratic optimisation problem. This formulation, called in this thesis problem **P2** is here solved using conjugate gradient (CG) algorithms on neuro–fuzzy systems. This solution includes the problem of optimisation in RBFs as a special case. It then becomes apparent(El–Zobaidi and Jaimoukha 1998) that one can extend the study of conjugate gradient to a general class of optimisation on orthogonal spaces known as *Arnodi's method* (Saad 1992). This is an orthogonal projection method, similar to Gram Schmidt method (Luenberger 1969). The detail account of the algorithms we used and their analysis for model reduction can be found in (Saad 1992).

We also presented a radically new approach to the problems of evaluating and optimizing the efficiency of these systems based on moduction reduction principles. The approach is based on the use of a family models that are again based on a quadratic values of the objective function. In choosing the best parameters in the model or the best data points that can best explain the systems, cost functional turns out to be the variance of the residuals as a function of the parameters or the data points in question.

This is how this chapter is organised. The first part in section(2.2), shows how model reduction lead to analysis of variance and the main properties the modelling framework are demonstrated. There is particular emphasize in Section (2.5.1) on the application of the conjugate gradient algorithms, why we choose it and the relationship with a general class of orthgonal projection algorithms known as *Arnodi process.* The rest of that section explores the importance of analysis of variance and orthogonal projection algorithms through various experiments. Section(2.6) is devoted to a standard problem in RBF regression affiliated with the control of complexity of solution of approximation problem known as *regularisation.* We show how this problem can be formulated as an important constraint optimisation problem in which the solution is restricted to lie in ball in Euclidean space. This section equally shows how the model reduction problem can also be formulated as a constraint problem. The rest of the section deals with a number of illustrative experiments. This chapter concludes with a section on clustering, its place in data reduction and the application of a number of clustering algorithms.

Throughout this chapter modern methods of analysing stochastic systems are used, but all the necessary algorithms are developed from scratch.

2.1.1 Model reduction problem

Thus, given an expansion of the form (1.5.12), we formulate two abstract sub–problems, usually stated in statistical terms (Davis and Vinter 1984) as:

1. **Data reduction:** Given data points

$$D_1, D_2, D_3, \ldots, D_N \tag{2.1.1}$$

decide on the hypothesis that a subset d of the data points:

$$D_N, D_{N-1}, D_{N-2}, \ldots, D_{N-d+1} = 0 \tag{2.1.2}$$

should be rejected.

2. **Model reduction:** Given a model characterised by a parameter vector

$$\theta_0,\ \theta_1,\ \theta_2,\ \theta_3,\ \ldots,\ \theta_p \tag{2.1.3}$$

decide on the hypothesis that a subset k of the parameters:

$$\theta_p,\ \theta_{p-1}, \theta_{p-2},\ \ldots,\ \theta_{p-k+1} = 0 \tag{2.1.4}$$

should be rejected.

Note that in the experiments in this chapter we have adopted the non–linear auto–regressive systems outlined in chapter (1) where: y_{k+1} is the output that is conceived to be a function of some lagged outputs $(y_k, y_{k-1}, \ldots, y_{k-n_y})$. The Radial basis expansion (1.5.12) which we saw in chapter (1) is then used to approximate the non–linear relationship. As far as model reduction is concerned, the systems is non–linear, but with respect to the regressive parameters, it is linear, Gaussian and have uncorrelated noise term.

Also note that, in the regression formulation in chapter (1) and this chapter, we have adopted the usual statistical notation of representing the data in abstract form D and the parameters similarly by θ. In the general process model formulation used in sections (1.5.1) and (1.5.2) of chapter (1), the data D represents values of the outputs $y(t)$ and of the long vector $x(t)$ at a number N of discrete "times" t_i and will be a matrix (as suggested by equation (2.1.1) or a (very) long vector

of dimension[1] $N \times (n+1)$. Similarly, the abstract parameter vector θ in principle stands for the weights λ_j, the RBF centres c_j, and any other parameters required such as the RBF width β, although it will often be restricted just to the weights λ_j. If all the aforementioned parameters were included the dimension p of θ would be $p = 2n_c + 2$ since there are, in general, two parameters for each RBF representing its centre and its weight as well as the bias λ_0 and the width β of the RBFs. We note in passing that, in particular case when Gaussian RBFs are used to represent fuzzy membership functions as discussed in section (1.5.3) of chapter (1), each RBF at node j may have its own width parameter β_j and that even more complicated Gaussian functions may be used.

The data reduction problem enables us to say that some of the data points do not contribute useful information towards determining the model parameters and may therefore be ignored. A classic example is provided by support vector machines, or other maximum margin classifiers, which usually only depend on the small fraction of key data points that determine the decision boundaries. In distinction, model reduction problem is concerned with limiting model complexity and thus avoiding the problem of over-fitting. In this case a classic example is the problem of determining the number of hidden units to be used in a neural network. Another example involving both sub–problems can be taken from feature selection in pattern recognition. This is more complicated since we may not only want to select features that ignore some or many of the data points[2], but may also want to select which data attributes(the components of the long vector $x(t)$) contribute to useful features[3], and decide which features are themselves useful in robustly determining the system's output[4]

2.2 Methods based on analysis of variance

When the problem has been rewritten in the linear form given in equation(2.4.6), analysis of variance (ANOVA) methods (see for example, (Draper and Smith 1998)) may be used both to select the model structure and for data reduction. To illustrate the approach, we assume that the errors:

---

[1]Recall that, in section (1.5.2) we assumed for simplicity a scalar output $y(t)$.

[2]As for example, edge features do in image processing.

[3]In image processing, a slightly contrived example might be how different colour channels are combined in the definition of an edge feature.

[4]If we continue with our image processing analogy, an example would be choosing strong edge features, say for object detection or recognition, so as to ignore distracting background cluster and noise.

$$\epsilon = (\epsilon_1, \ldots, \epsilon_N) \tag{2.2.1}$$

in (2.4.6) are normally distributed with zero mean and covariant matrix $\sigma^2 I$.

For the model reduction, we focus our attention on the number of weights λ_j required in the RBF expansion (1.5.12) which determines the number of hidden units in our network. Thus, we include only these weights in the parameter vector θ of equation (2.4.8) and partition θ into two parts,

$$\theta = (\theta_I, \theta_{II}) \tag{2.2.2}$$

where the first part,

$$\theta_I = (\theta_0, \theta_1, \ldots, \theta_{p-k}) \tag{2.2.3}$$

represents the weights we expect to retain, and the second part,

$$\theta_{II} = (\theta_{p-k+1}, \ldots, \theta_{p-1}, \theta_p) \tag{2.2.4}$$

those we seek to reject.

We similarly partition the matrix $A = (A_I, A_{II})$ and write the linear form (2.4.6) as:

$$Y = A_I\theta_I + A_{II}\theta_{II} + \epsilon. \tag{2.2.5}$$

Thus, if we refer to our RBF regression formulation, we can see that Y is $N \times 1$, A_I is $N \times p - k + 1$, and that A_{II} is $N \times k$. Since in this case with just one adjustable weight per RBF plus the bias λ_0, $p = n_c$, A_I is in fact a matrix of dimensions $N \times n_c - k + 1$.

The weights λ are linear, so least squares estimate of the parameter θ is given, as usual, by:

$$\hat{\theta}_{ls} = (A^T A)^{-1} A^T Y. \tag{2.2.6}$$

The sum of squared errors is defined as:

$$SSE(\hat{\theta}_{ls}) = \sum_{i=1}^{N} (Y_i - \hat{Y}_i)^2, \tag{2.2.7}$$

where,

$$\hat{Y}_i = a_{i0}\hat{\theta}_0 + a_{i1}\hat{\theta}_1 + \ldots + a_{ip}\hat{\theta}_p. \tag{2.2.8}$$

It may be written in terms of a *projection*:

$$SSE(\hat{\theta}_{ls}) = Y^T(I - H)Y, \tag{2.2.9}$$

where:

$$H = A(A^TA)^{-1}A^T \tag{2.2.10}$$

is the matrix that projects into the estimation space defined by the columns of A. It follows from this that the error or residual vector ϵ representing the difference between the predicted $\hat{Y}$ and the observed output Y, is given by:

$$\hat{\epsilon} = Y - \hat{Y} = (I - H)Y. \tag{2.2.11}$$

It turns out that $\hat{\epsilon}$ is orthogonal to the predicted value $\hat{Y}$. Furthermore, the covariance

$$cov(\hat{\epsilon}) = E\left\{[\hat{\epsilon} - E(\hat{\epsilon})]\,[\hat{\epsilon} - E(\hat{\epsilon})]^T\right\} = (I - H)\sigma^2, \tag{2.2.12}$$

from which we may obtain the correlation ρ_{ij} between $\hat{\epsilon}_i$ and $\hat{\epsilon}_j$

$$\rho_{ij} = \frac{-h_{ij}}{(1 - h_{ii})(1 - h_{jj})}. \tag{2.2.13}$$

Equation (2.2.13) follows since the variance $var(\hat{\epsilon}_i)$ of the i^{th} residual is $(1 - h_{ii})\sigma^2$ and the covariance $cov(\hat{\epsilon})$ between the i^{th} and the j^{th} residual is $-h_{ij}\sigma^2$,where:

$$h_{ij} = a_i(A^TA)^{-1}\,[a_j]^T$$

are the elements of the projection matrix H written in terms of a_i and a_j denoting the i^{th} and the j^{th} row of the data matrix A., respectively.

It may further be shown that:

$$h_{ij} = \frac{1}{N} + (a_i - \bar{a})(A^TA)^{-1}(a_j - \bar{a})^T, \tag{2.2.14}$$

where $\bar{a}$ is the mean of a_i. The second term in (2.2.14) represents the equation of an ellipsoid centred at $\bar{a}$ for $i = j$. If the term $(1/N)$ in (2.2.14) is dropped from the right–hand side of the equation, and the remaining term is multiplied by $(N - 1)$, the quantity

$$(N - 1)(a_i - \bar{a})(A^TA)^{-1}(a_i - \bar{a})^T \tag{2.2.15}$$

is called the Mahalanobis distance from a_i to the centroid of the data $\bar{a}$.

Usually, σ^2 is estimated from s^2, the sum of squared residuals, i.e. if we recall that

$$s^2 = (\hat{\epsilon})^T(\hat{\epsilon})/(n-p-1). \tag{2.2.16}$$

Although we made the assumption that the error ϵ_i in equation (2.4.6) were not correlated, the residuals $\hat{\epsilon}$ are correlated and have variances that depend on the locations of the data points. It is a common practice to scale or "Studentise" the residuals so that they all have the same variance. In the equation that follows, the scaled residual, r_i, has a Student's t distribution with $(n-p-1)$ degrees of freedom.

$$r_i = \frac{\hat{\epsilon}}{(\hat{\sigma}_i)\sqrt{1-h_{ii}}} \tag{2.2.17}$$

where[5]

$$\hat{\sigma}_i = \frac{\hat{\epsilon}^2}{(N-p-2)} - \frac{\hat{\epsilon}_i^2}{(N-p-2)(1-h_{ii})}. \tag{2.2.18}$$

A useful statistic is then the so-called coefficient of correlation, R^2, defined as:

$$R^2 = \frac{\sum_{i=1}^{N}(\hat{Y}_i - \bar{Y})^2}{\sum_{i=1}^{N}(Y_i - \bar{Y})^2}, \tag{2.2.19}$$

where $\bar{Y}$ is the mean of the observations Y. R^2 measures the "proportion of total variation about the mean $\bar{Y}$ explained by the regression". It lies in the range $[0,1]$ and is often expressed as a percentage.

2.3 Gaussian time series approximation

In the data sets provided, we assume that n is the number of sources. It should be noted that in the electricity board $n = 10,000,000$; in the insurance claims n runs into thousands and in the IBM message passing protocol $n = 32$, representing the number of CPUs that can run the MPI (data communication) tasks. In general $N(t)$ denote the total number of requests coming in a small time interval $(t-1,t]$.

$$N(t) = \zeta_1 + \zeta_2 + \ldots + \zeta_n \tag{2.3.1}$$

is the sum of independent Bernoulli random variables $\zeta_1, \zeta_2, \ldots, \zeta_n$, where $\zeta = 1$ if customer is active in the time interval $(t-1,t]$ and $\zeta = 0$ if customer is not active in that time interval. Suppose the probability of being active is constant p and not

[5]Again recall that, thanks to the bias λ there are $p+1 = n_c+1$ parameters.

active is $q = 1 - p$. This kind of Binomial representation is also used in the analysis of queueing systems, traffic, ATM switching problems (Gibbens Kelly and Key 1995, Hui 1988) to estimate customers demand and bandwidth prediction.

Then the random variable $N(t)$ has a Binomial distribution

$$B(k; n, p) = P[N(t) = k] = \begin{pmatrix} n \\ k \end{pmatrix} p^k q^{n-k}, k = 0, 1, \ldots,$$

Notice that the number of Bernoulli trials n is large. Now suppose that the probability of success p is relatively small so that each success is a *rare* event and np is of moderate size. Then it is a good approximation to have

$$P(k; n, p) = P[N(t) = k] \approx \frac{\lambda^k}{k!} e^{-\lambda}, k = 0, 1, \ldots,$$

where $\lambda = np$ is the average number of successes. We have $B(0; n, p) = (1-p)^n$ and

$$B(0; n, p) = (1 - \frac{\lambda}{n})^n.$$

$$\log B(0; n, p) = n \log(1 - \frac{\lambda}{n}) = -\lambda - \frac{\lambda^2}{2n} - \ldots$$

using Taylor's expansion, so that for large n

$$B(0; n, p) \approx e^{-\lambda}.$$

And from induction we arrive at a Poisson approximation.

We note here that when the Bernoulli parameter p and the number of customers satisfy some conditions the demand process can be approximated as a Poisson process. We will revert to how we move from Poisson approximation to Gaussian time series later. The modelling assumptions which have lead us to Binomial are that the conditions of the demand process remain constant in a finite interval $[0, T]$, that the non–overlapping time intervals $[t-1, t]$ are stochasticllay independent in the usual sense (*Davis*1993) and also in the sense that information concerning the number of demands by one customer reveals nothing about the other customer. Thus, in a particular interval we have n number of demand slots, each slot is empty or it contains at least one unit of workload (or demands), and we call the two possibilities failure or success, respectively. Unfortunately, the number of success is not the same as the amount of demand units, since a customer may request more than one unit of demand in a single time interval $[t-1, t]$. However, it is common to make a further assumption that the probability of a customer making more than a unit

BIBL. LONDIN. UNIV.

of demand is in the limit very small. In this case the probability of having exactly k units of demand is $B(k;n,p)$. If instead of the interval of unit length, we have interval $[t-\tau,t]$ of length τ, the Bernoulli trials with same probability p of success, but the number of trials is the integer nearest to $n\tau$ rather than n.

We will note that the modelling assumption used applies, in *central limit theorems (CLT)*, for the demand process being Binomial and Markov. Suppose the superposition process $\{N(t)\}$ is not Binomial, then the process would not be Markov. To verify,we take the modified interval $[t-\tau,t]$ with length τ so large that our assumption of a customer making more than a unit of demand is no longer negligible. This is actually a transient problem in a non–Markovian setting and PDP theory deals with this.

For this purpose, consider n as independent renewal processes $\{N_k(t), t>0\}, k=1,2,\ldots,n$, where k is the k^{th} customer as in the intoduction and $N_k(t)$ is the number of claimers of a certain claim type k that arrive during a time period $(0,t]$. Following the technique of chapter(7), the time interval $(0,t]$ is partitioned into t sub–intervals[6] Let $N(t)=\sum_{k=1}^{n} N_k(t)$ be the sum of these arrivals. Denote by $\{F_k(t)\}$ the probability distribution function of the inter renewal time between consecutive arrivals of type k and denote by α_k the first moment of $\{F_k(t)\}$. Let τ_k be the elapse time until the next arrival in the process $\{N_k(t)\}$. Also, let $\tau=\min(\tau_1,\tau_2,\ldots,\tau_n)$ be the waiting time until the next renewal in the superposition process $\{N_k(t)\}$. It is true that for any renewal process $N_k(t), k=1,2,\ldots,n$, each with waiting times τ_k. And if each process seems to be in progress for a very long time, we have by (??) holds. In addition, if we assume t at a large value and also vary n at large values

$$P[\tau_k \le t] \approx \frac{t}{\alpha_k}.$$

Therefore,see (Tjims 1986), the superposition process has waiting time $\tau=\min(\tau_1,\tau_2,\ldots,\tau_n)$

$$P[\tau_k > t] = P[\tau_1 > t]\,P[\tau_2 > t]\ldots P[\tau_n > t].$$

$$P[\tau_k > t] \approx \left(1-\frac{t}{\alpha_1}\right)\left(1-\frac{t}{\alpha_2}\right)\ldots\left(1-\frac{t}{\alpha_n}\right) \approx e^{-\lambda t},$$

where

$$\lambda = \frac{t}{\alpha_1}+\frac{t}{\alpha_2}+\ldots+\frac{t}{\alpha_{n-1}}+\frac{t}{\alpha_n}.$$

[6] $t \equiv$ integer nearest to t.

Let

$$\phi(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}$$

be the normal density function whose distribution is denoted as $\Omega(x)$. If we now define $\mathbf{S}_n$ as the number of successes in n Bernoulli trials with probability p of success. If α and β are integers $\alpha < \beta$, then the event $\alpha \leq \mathbf{S}_n \leq \beta$ has probability

$$P[\alpha \leq \mathbf{S}_n \leq \beta] = B(\alpha; n, p) + B(\alpha+1; n, p) + \ldots + B(\beta-1; n, p) + B(\beta; n, p). \quad (2.3.2)$$

One version of the *central limit theorem* states that as follows.

Lemma 2.3.1 *If α and β vary so that $lx_\alpha^3 \to 0$ and $lx_\beta^3 \to 0$, then*

$$P[\alpha \leq \mathbf{S}_n \leq \beta] \approx \Omega(x_{\beta+0.5}) - \Omega(x_{\alpha-0.5}),$$

where $l = (npq)^{-\frac{1}{2}}$ and $x_0 = (\theta - np)l$.

The motivation leading to this lemma was derived from the fact that the sum in (2.3.2) may involve many terms in which direct evaluation becomes impractical. Fortunately, whenever n is large, the Gaussian distribution function enables us to compute an approximation to the probability (2.3.2). In terms of this lemma we can rewrite the Binomial distribution as

$$B(k; n, p) \approx l\phi(x_k),$$

which goes back to De–Moivre in the 17^{th} century. Figure (2.1) illustrates the lemma in the case $n = 20, p = 1/5$ where variance npq is only 3.2. Even in this hypothetical situation where the variance npq is small, the approximation is fairly good.

This lemma is historically the first limit theorem in stochastic analysis. The key basis of the *central limit theorem*. That is what we have based our approximation of the data analysis part of this thesis on. We use this Gaussian approximation on the electricity, insurance claims and the tea preference data because the number of customers n involved is large, making the variance npq of the process relatively large. It is common for other modellers to use the lemma even when npq is small, and in such cases it will be advisable to state the estimate of the approximation error of moving from the discrete Binomial to a continuous Gaussian process.

For example, see chapter(8) consider the message passing case the number of cpu tasks is $n = 400$ and $p = 0.5, \alpha = 45, \beta = 55$. Now $P[45 \leq \mathbf{S}_n \leq 55]$ will have

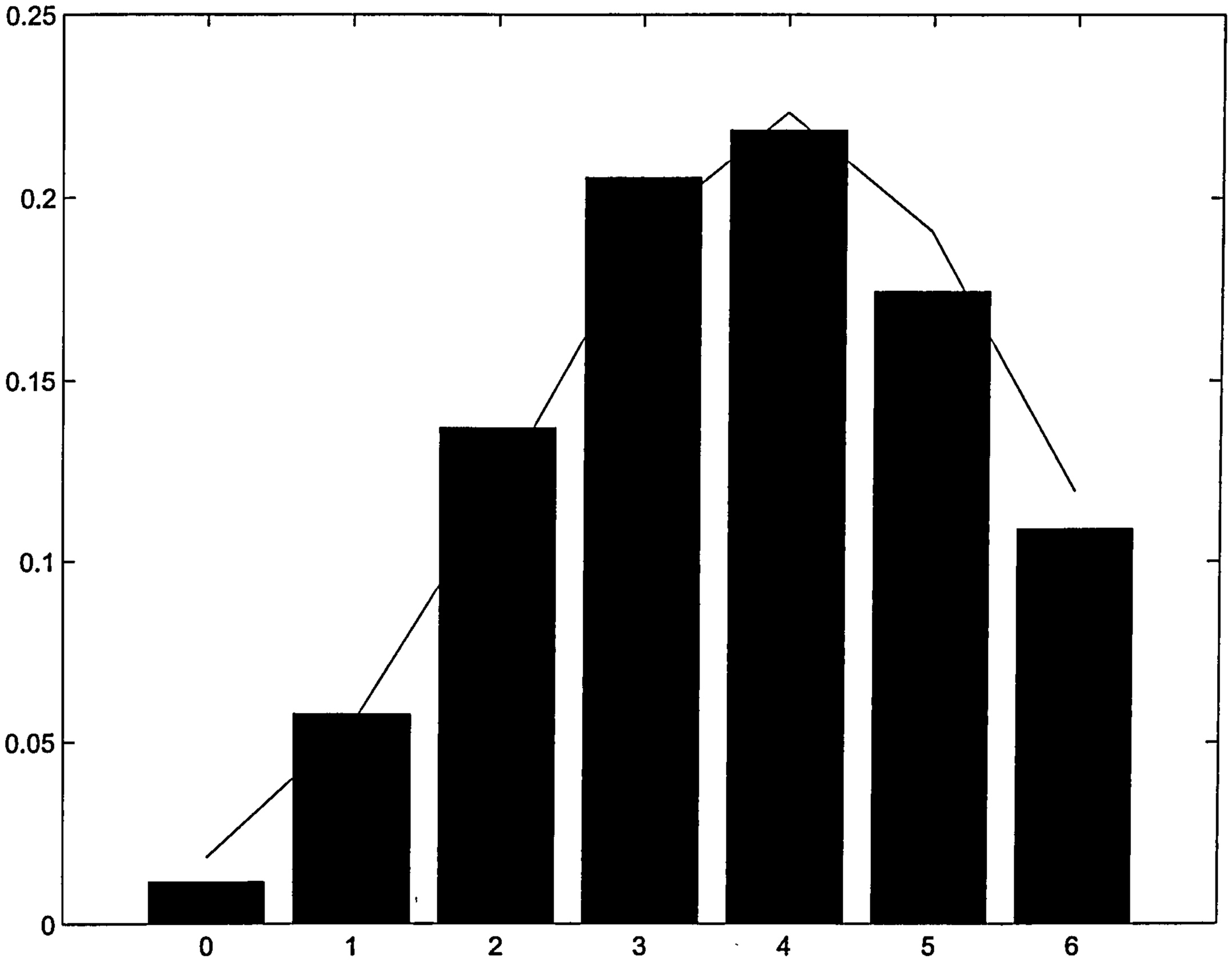


Figure 2.1: Gaussian approximation to a Binomial distribution

the interpretation: in the sample interval $(t-1,t]$ the probability that among the n tasks running those with work load that deviates from 50 by at most 5. Thus, $l = (100)^{-\frac{1}{2}} = 0.1, x_{\alpha-0.5} = -14.45, x_{\beta+0.5} = -15.55$. From the cumulative Gaussian distribution $\Omega(x_{\beta+0.5}) - \Omega(x_{\alpha-0.5}) = 1.2534e-047$ and from the sum

$$P[\alpha \leq \mathbf{S}_n \leq \beta] \approx l[\phi(x_\alpha) + \phi(x_{\alpha+1}) + \ldots + \phi(x_{\beta-1}) + \phi(x_{\beta-1})] \qquad (2.3.3)$$

the probability is $1.1501e-047$. The error between the two approximations is quite small. It turns out that in most cases the error committed in using the sum rather than the integral is quite small.

For the electricity data, the set up is that the $n = 10,000,000$ customers are to use intermittently the electric power, and we are interested to predict the total load to be expected. A crude approximation is to imagine that at any given time each customer has the same probability p of requiring a unit of power. If they are

operating independently, the probability of exactly k customers requiring power at the same time should be $B(k; n, p)$. Currently, short–term forcasts are done based on data sampled over 60 minutes and the sample period is a single minute.

2.4 Computer communication among nodes

For message passing on IBM computer server nodes, the set up is that $n = 32$ CPU could at any sampling interval makes an I/O requests to send a message to another node. For simplicity we consider just two computing nodes A and B. There are 32 tasks running on each machine which want to exchange information with the other 32 tasks on the 32 CPUs on node B. It will be too expensive to design 32 I/O buses on each node with a dedicated channel to the other node. However, it will be better to choose the number of I/O ports or buses , N per node so large that, under message passing operational conditions, only one out of every thousands of I/O requests fail to find a I/O bus to use. The time taking to send a message to the remote node and gets an acknowledgment back is a function of the message size, the link bandwidth and the latency between the two nodes. This is explained in chapter(8).

Let τ be the sampling interval in a message passing study and n the number of CPU tasks. The physics of making I/O request is equated to that of n trials with probability p that an I/O bus will be required by a CPU. We have, then, 32 Bernoulli trials with say , $p = 1/200$. This is just one example of request rate.

$$S_n = x_1 + x_2 + \ldots + x_n,$$

where x_i is a Bernoulli variable with parameter p.

As indicated in the introduction, our demand process is a sequence $X_0, X_1, \ldots, X_t$, where $X_t = S_n$ is the total demand in the time interval $(t-1, t]$, with sampling time τ. Our problem remains: given the above sequence up to time t, what is the predicted demand at time $t+1$.

In actual application the buffered memory area where the adaptor uses for DMA communictaion is pinned during the duration of the message passing protocol operation. Locking these pages is undesirable since it defeats the flexibility of demand paging by the Kernel operational system, but it solves the performance problem of the adaptor not reading the Data from the Disk with high latency. In addition, to predict the effective throughput of the system, the following set up is required. It is quite possible to measure the demands from which we can obtain a time series data

to make prediction for the future.

Using IBM queue simulation, we simulate the protocol with two CPUs (i.e. $n = 2$) running on the send side and 1 CPU on the receive side for one direction transfer. The two CPUs are running MPI jobs and one adaptor transmitting the messages from the pinned memory area and releasing the messages from the pinned areas as soon as the receive side acknowledges the messages. Using simulation we are capable of predicting the best size of packets to be acknowledged rather than acknowledging every packet. This is an example of two MPI tasks passing data to a single node. It is an example of multiple tasks sending to one task. There are other examples in the client–server model with this type of queueing structure. In this experiment messages of $4kBytes$ arrive at user space 1 at inter–arrival times that is exponentially distributed with mean 5 based cycles. Upon arrival messages are copied from the user space to a pinned area with mean memory–to–memory copy time that is 2 based cycles. In addition, there are some additional delay on the interfaces between the user buffer and the pinned area and that of the pinned area and the hardware adaptor. These are incorporated in the modelling using some other distributions. In chapter (8) we shall discuss the kind of experiments that are done to estimate the transit time. However, it is important to mention here that the process of transferring data from the user space to a pinned memory area involves a series of software operations that prepares the message to be transmitted through a DMA process into the adaptor link to the other side of the node. A similar set up is followed in the user space of the second task except that the inter–arrival times are exponentially distributed with mean of 30 based cycles and message sizes 4 times as big as those in the pinned area of the first task. We also introduce some processing delays on the link and the other node. These are given some distributions. Additional assumptions are made such as: 9% probability that a packet is not delivered. This process set up is then simulated on an IBM queueing simulator and data is collected of the number of packets X_τ on adaptor buffer at the instant τ when a packet arrives. τ is a random variable depicting the waiting–time until a renewal occurs when a packet arrives and at that instant amount of untransmitted packets is X_τ, which again is a random variable, depicting the queue length at renewal times $\tau = 12.1, 15.4, 17.9, 23.5, 26.4, 29.0, 31.5, 34.0, 37.2, 39.5$ with inter–renewal times $3.3, 2.5, 5.5, 2.9, 2.6, 2.5, 2.5, 3.2, 2.3, 5.4$, base cycles.

2.4.1 Batching for model reduction

Consider a queue system of messages over a time interval $[0, T]$ and let N_t be the number of packets in the queue at time t and let τ_t denote the time elapsed since last change of state of N_t. For each $t \in [0, T]$ let n_t the number of change of states that has occurred by time t. Then $X_t = (n_t, \tau_t, t)$ is a renewal process with inter–renewal time $\min(A, D)$, where A and D are the inter–arrival time and inter–departure time of packets on the channel, respectively. Without abusing notation, suppose, (n_i, t_i) is the (number of packets in the system, inter–renewal time) at the time of the i^{th} renewal. In general $\sum_{i=0}^{\infty} t_i = T$. Then n_i is the queue length at the i^{th} departure or arrival.

Let n_T denote the total number of arrivals or departures that have occurred in the systems during $[0, T]$. Therefore, the time–weighted–average queue length is

$$\bar{N} = (1/T) \sum_{i=1}^{N_T} n_i t_i = \sum_{i=1}^{N_T} n_i (t_i/T). \tag{2.4.1}$$

In one MPI simulation described above, we set the time interval $[0, 50, 000]$, where T is $50, 0000$ base cycles. For each replication r, the time interval is divided into 50 sub–intervals and the estimate N_{rj} with $r = 1, 2, \ldots, 10$, is computed over each interval $[1000(j-1), 1000j]$. The results provide batches $(N_{r1}, N_{r2}, \ldots, N_{r50})$ for each replication.

In practice we average all the batch means within each replication to obtain a replication average. However, a goal for model reduction is to identify the trend in the data due to initialisation bias and find out when it dissipates. One approach(which will be done in the next chapter reduces to the question of estimating the stationary mean of a process from its transient response) is addressed. Another model reduction approach is deletion of data points that do not contribute much to the mean. Hence, the problem can be formulated as a data reduction problem as earlier, where the contribution of each batch is measured against replication mean, what is called *ensemble mean*. Figure (2.2) shows the replication mean for 50 batches and replication mean, what is called *ensemble mean*. Figure (2.3) shows it for 500 batches.

So, we can see that the data consist of results for a single realization (i.e., one replication) of a certain batch means at certain discrete time.

We now create a regression matrix data $(y, X), y = y_{k+1}, X = (y_k, y_{k-1}, y_{k-2}, y_{k-3})$ of four lags using vector of 500 batches and also that of 5000 batches as input.

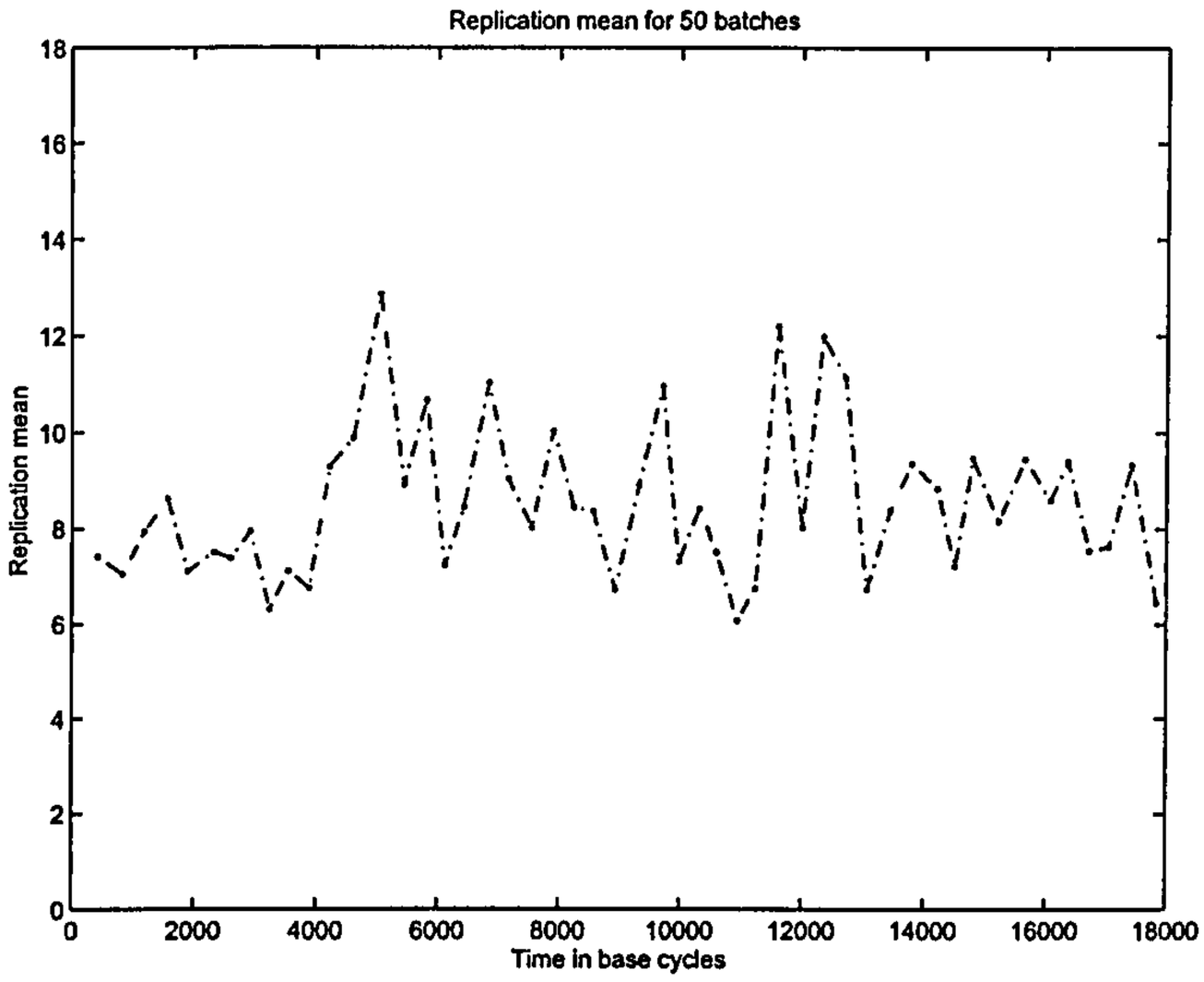


Figure 2.2: Replication mean across batches

Examine the first 10 rows of y and the corresponding rows of the lags 500 batch case:

y_{k+1}	y_k	y_{k-1}	y_{k-2}	y_{k-3}
3.5738	7.7604	7.7604	7.7604	7.7604
4.2863	3.5738	7.7604	7.7604	7.7604
4.6625	4.2863	3.5738	7.7604	7.7604
5.7694	4.6625	4.2863	3.5738	7.7604
7.6675	5.7694	4.6625	4.2863	3.5738
9.3030	7.6675	5.7694	4.6625	4.2863
7.9937	9.3030	7.6675	5.7694	4.6625
8.5097	7.9937	9.3030	7.6675	5.7694
8.2106	8.5097	7.9937	9.3030	7.6675
6.2989	8.2106	8.5097	7.9937	9.3030

The value "7.7604" in the resulting data matrix indicates the presence of a missing observation which we replace with the mean before any analysis. The problem structure takes the form

$$y_{k+1} = \mu + \theta_1 y_k + \theta_2 y_{k-1} + \theta_3 y_{k-2} + \theta_4 y_{k-3} + \epsilon_k. \tag{2.4.2}$$

The data reduction problem is to choose data points or equivalent batches the best describe the process. Note here that the number of parameters $(\mu, \theta_1, \theta_2, \theta_3, \theta_3, \sigma)$

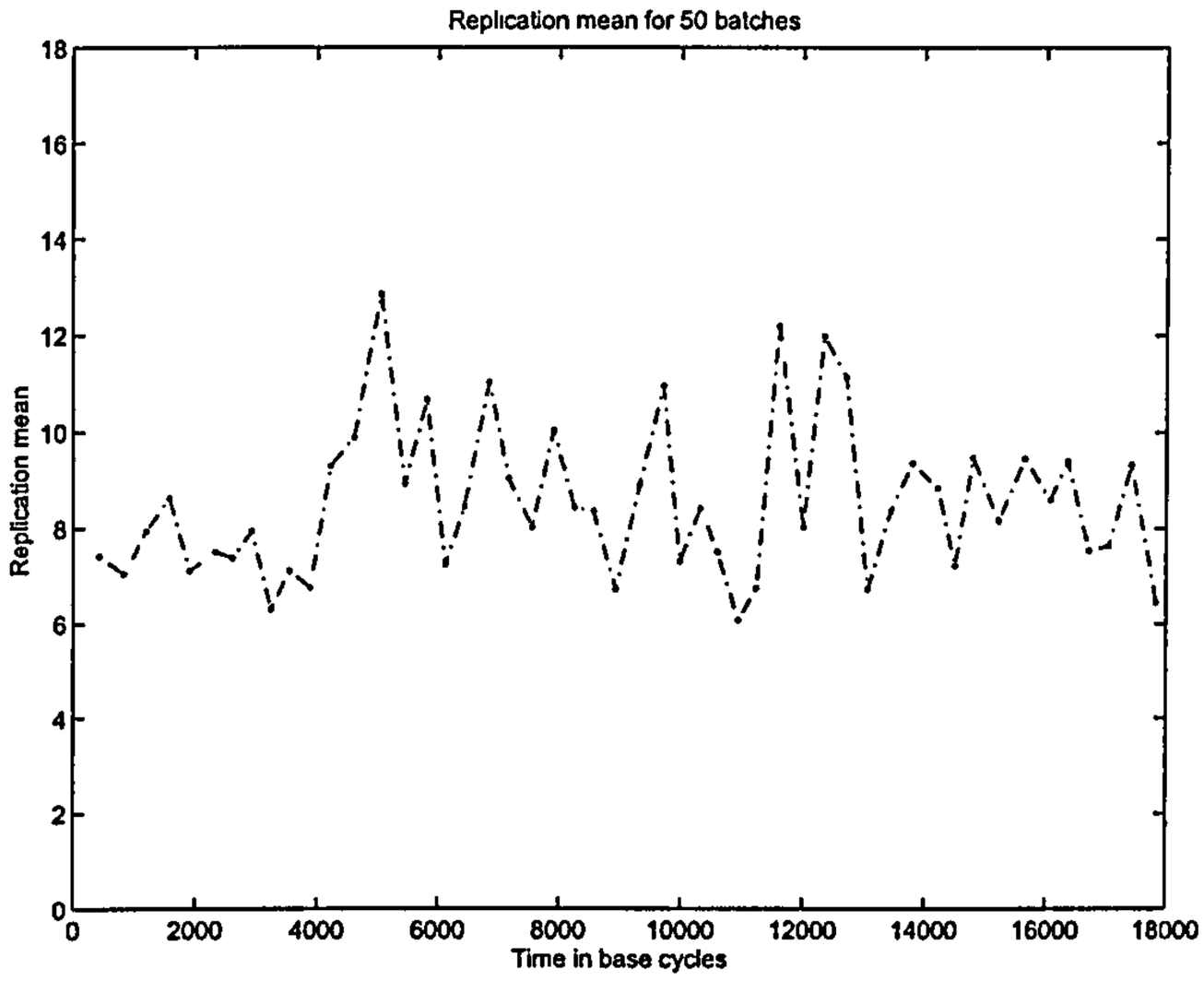


Figure 2.3: Replication mean across batches

for optimisation is 6, where σ^2 is variance.

Parameter	estimate	Standard Error	T Statistic
μ	2.2126	0.29523	7.4943
θ_1	0.86426	0.043335	19.9437
θ_2	−0.20354	0.063023	−3.2296
θ_3	0.14768	0.06322	2.3359
θ_4	−0.093765	0.045632	−2.0548
σ	2.3698	0.14164	16.7311

We use two different types of optimisation algorithms. The first is shown above. The second Algorithm produces $\mu = 2.2206, \theta_1 = -0.2026, \theta_2 = 0.1459, \theta_3 = -0.2026, \theta_3 = 0.8644, \sigma = 2.3699$). There are several ways of solving this optimisation problem, including the method of conjugate gradient. The set of procedures for optimisation is explained below. We computed the F–statistics of 170, with zero probability, which is the hypothesis test that all the all θ are zero and the probability associated with this hypothesis. We found that the model accounts for 58% of variability in batches. It is therefore highly unlikely that all the *thetas* are zero. This implies that the queue length of packets through this communication channel over period of $50,000$ base cycles, measured in micro seconds. The plot in figure (2.4) shows points that are deleted which are out of range.

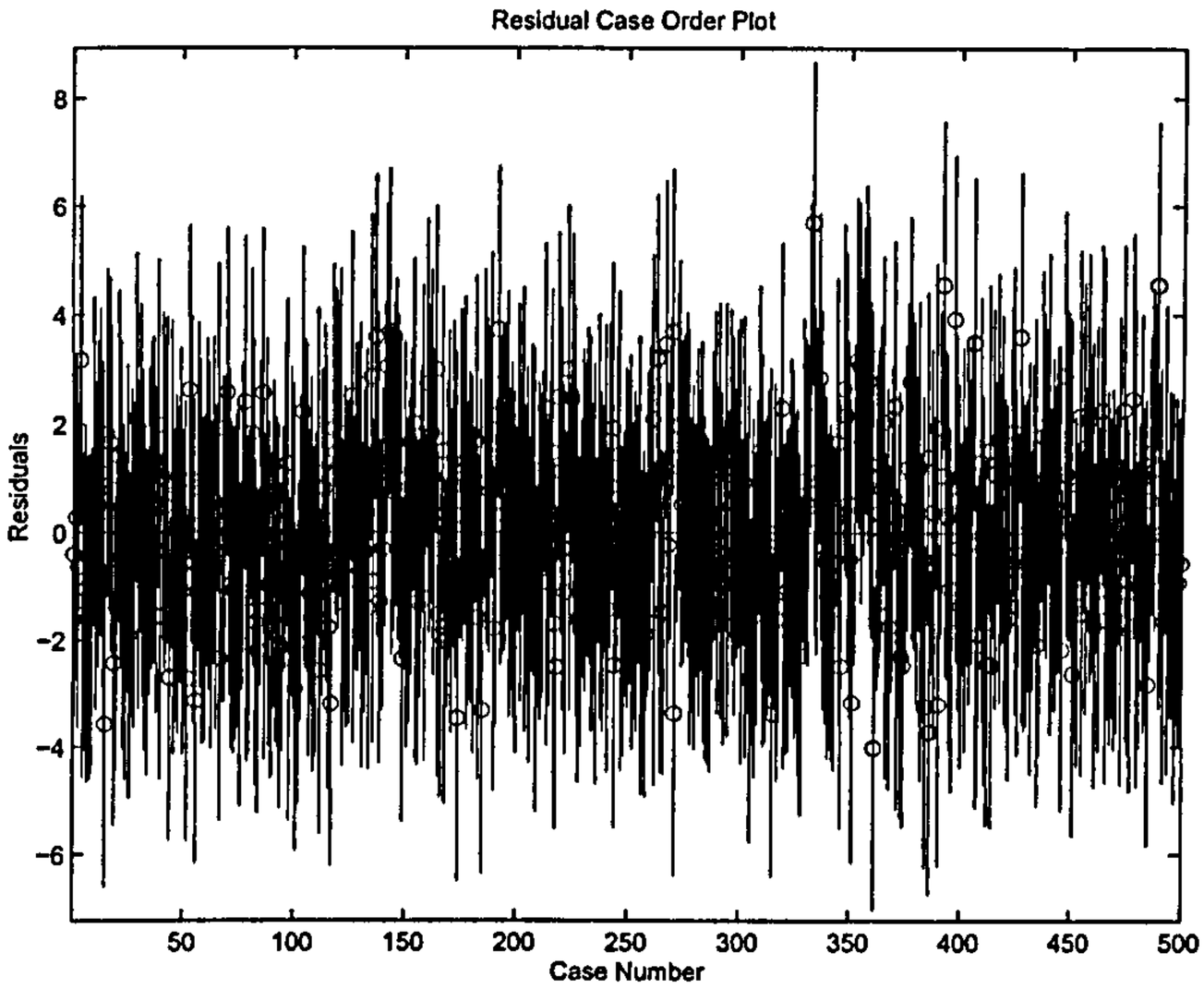


Figure 2.4: Replication mean across batches

2.4.2 Selecting important data points

To illustrate the kind of data reduction , a small test example from the data mining inventory at UCL is used. The data set has four attributes and one response.

Response	Attributes			
y	x_1	x_2	x_3	x_4
0.909	1.287	0.984	0.987	1.046
1.252	1.281	1.078	1.064	1.081
0.947	0.787	1.061	1.007	1.051
1.022	0.796	1.013	1.012	1.046
...	...	...	...	...

Only the first four data entries are given above. The full data set consists of sixteen such entries describing the total annual demand of rubber over a number of years, along with attributes such as car production, gross national product, etc., which we might expect to affect the annual demand for rubber, although we cannot be certain of this and, more importantly, we do not know whether they are sources of independent information. It is a small example for illustrative purposes only but, within the context of the data sets considered in this thesis, it could be regarded as similar to the sort of data produced in a demand processes, although it is probably less variable.

We use Gaussian radial basis functions, defined by:

$$\phi_j = \exp\left(-\frac{(x-c_j)^T(x-c_j)}{\beta^2}\right), \tag{2.4.3}$$

where c_j is the centre of the function at node j and β is the width constant.

In this illustrative experiment we developed Matlab code to compute the statistical properties of the RBF model represented by equation (1.5.12). To do so, we use as a standard trick (Zhu and Billings 1996) to rewrite (1.5.12) in a form reminiscent of linear statistics problems by introducing a matrix A with elements constructed from the RBF:

$$a_{ij} = \phi(||x(t_i) - c_j||) \tag{2.4.4}$$

with, in order to include the bias,

$$a_{i0} = 1. \tag{2.4.5}$$

With a scalar output, $y(t)$, as assumed above and data points at "times" $t_1, t_2, \ldots, t_N$, A is an $N \times (n_c + 1)$ dimensional matrix. Given the matrix A, we can rewrite the RBF expansion (1.5.12) in the linear form:

$$Y = A\theta + \epsilon, \tag{2.4.6}$$

where

$$Y = (y_1, \ldots, y_N)^T \tag{2.4.7}$$

is the dependent random variable, and

$$\theta = (\theta_0, \ldots, \theta_{n_c})^T \tag{2.4.8}$$

is the parameter vector. Note that in this case where the centres c_j and width β of the RBFs are fixed, θ represents only the weights $\lambda_j, 0 \leq j \leq n_c$.

Since, as stated above, we are interested in data reduction and model reduction, we wish to determine in the illustrative experiments whether some data entries do not provide useful information for determining the model and whether some basis functions do not have a significant effect on the system's output. The former should be excluded from the training data used to define the model parameters whilst the latter should be removed from the model as their parameters are likely to be poorly determined and they have weak explanatory power. In the fuzzy logic setting, this would mean that some *antecedent rules* do not have a strong effect on the *consequent*

part of the system, i.e. that the rules would fire frequently without effect. It is also important to note that the data D should be representative of the problem or process of interest. Any conclusions we draw are, of course, only as good or reliable as the data on which they are based.

In statistical terms, having rewritten the problem in a linear form as in equations (2.4.6) — 2.4.8), we need to find the quantitative relationships between the columns of A and the response, Y and between the parameters θ and the response Y. This will enable us to:

1. understand which data points have the most effect,
2. predict future values of the system's response when only the current and previous values of the system's inputs and outputs are known,
3. compute the degree of variability in the model,
4. select the basis functions to include in the model.

In order to illustrate the difficulties that can occur if one tries to guess the model complexity and parameters, we first consider a model with sixteen radial basis functions, each with a small value of $\beta = 1$, as hidden units and assign their centres so that there is essentially one RBF to each data item. We create such a network by taking input matrix X and target vector Y in the data points, and choosing the value $\beta = 1$. We do this by creating as many radial basis neuron as there are input vectors in X, which we know is 16. We now have 16 neurons, each with center c_j equal to the vector at position j^{th} of the input matrix X.

If we present an arbitrary input vector z into the network, each of the 16 neurons j will fire activated signal according to how close the input vector is to its centre c_j. β provides us with the mean of adjusting the sensitivity of the neurons to inputs in such a way that if a neuron has $\beta^2 = 10$ it will output 0.5 for any input that is a distance of $8.326 (= 0.8326\beta^2)$. As we might anticipate from the fact that the number of hidden units is equal to the number of data entries[7], the model is unnecessarily complicated.

We shall see from application of orthogonal projection method in section (2.5.2) that four RBFs are sufficient as shown in figure (2.9), although to ensure that they

[7]We only anticipate this because, given its nature, we expect, as remarked earlier, that there will be correlations or redundancies in the data.

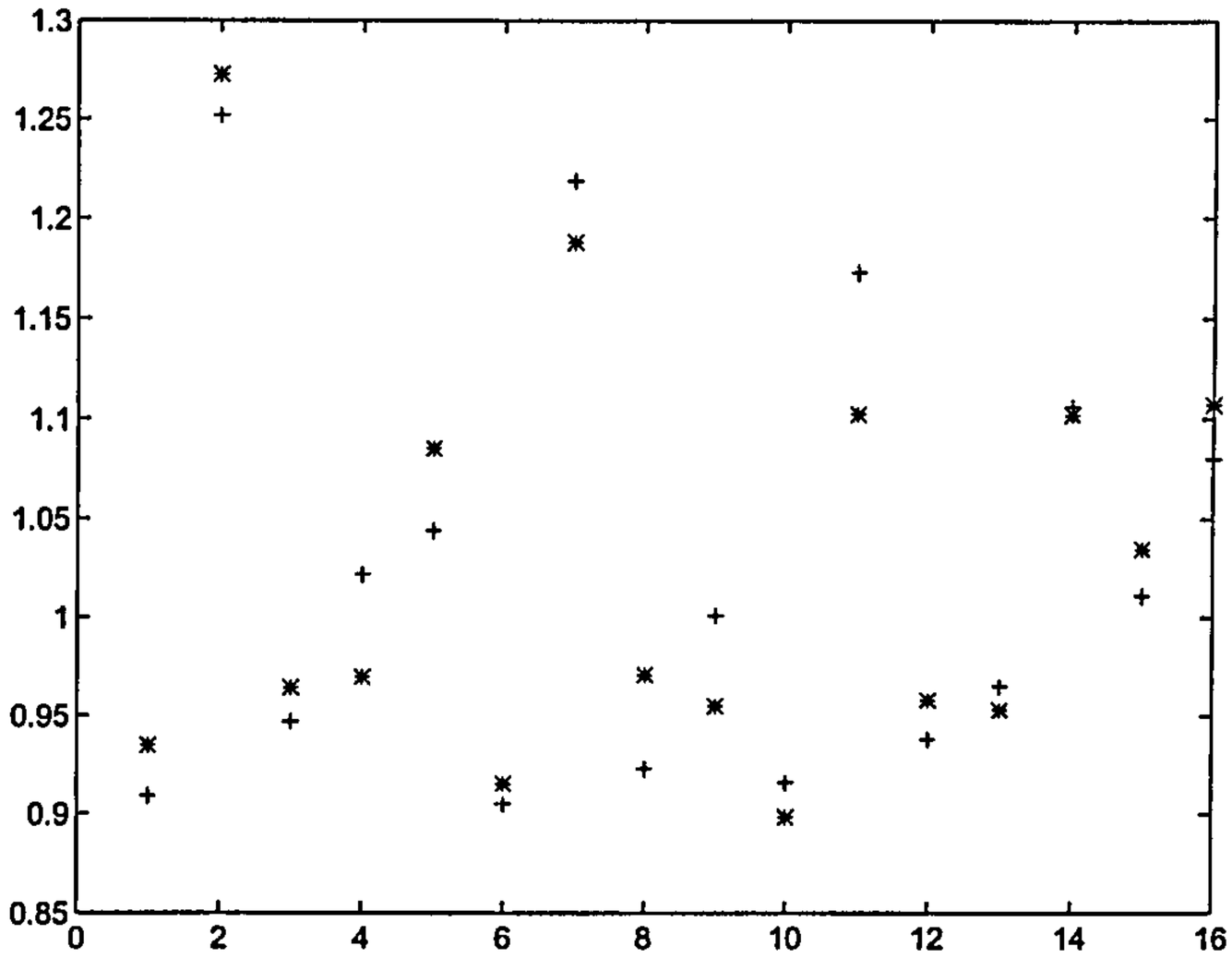


Figure 2.5: The 4 fuzzy rules do not overlap

sample the data sufficiently, they are wider with $\beta = 1.2$. It is interesting to note in passing that we know from the theory of fuzzy logic that, if we have a small number of RBFs that do not overlap, then the fuzzy rules associated with the data will not describe the data well. This can be illustrated by choosing four RBFs, but with a small value of $\beta = 1$ which leaves the functions isolated. The response for such a network with $\beta = 1$ is given in figure (2.5), with the input data values y denoted by a $'+'$ and the estimates obtained from equation (2.4.6) with $'*'$ markers. Such a network does not estimate the data points correctly.

However, when we increase the number of RBFs or hidden units to about 10 while maintaining the width $\beta = 1$, the fit was accurate, as shown in figure (2.6). A simple exercise with the first few data entries strongly suggests that with $\beta = 1$ the RBFs will overlap. Such a network, of course, describes the data well, as illustrated in figure (2.6), with the actual data values y denoted by a $'+'$ and the estimates obtained from equation (2.4.6) with $'*'$ markers and the networks give the right outputs.

This illustrates the kind of problems that can occur when trying to guess the model complexity and parameter values, in particular, when data is scarce and we cannot check (as here) on the model's predictive power on independent test data not used in the model fitting or training stage. In addition, if the model shown in figure (2.6) with 10 RBFs were interpreted as a set of fuzzy rules, the rules would be much less intelligible to a human user a small set based on just four RBFs.

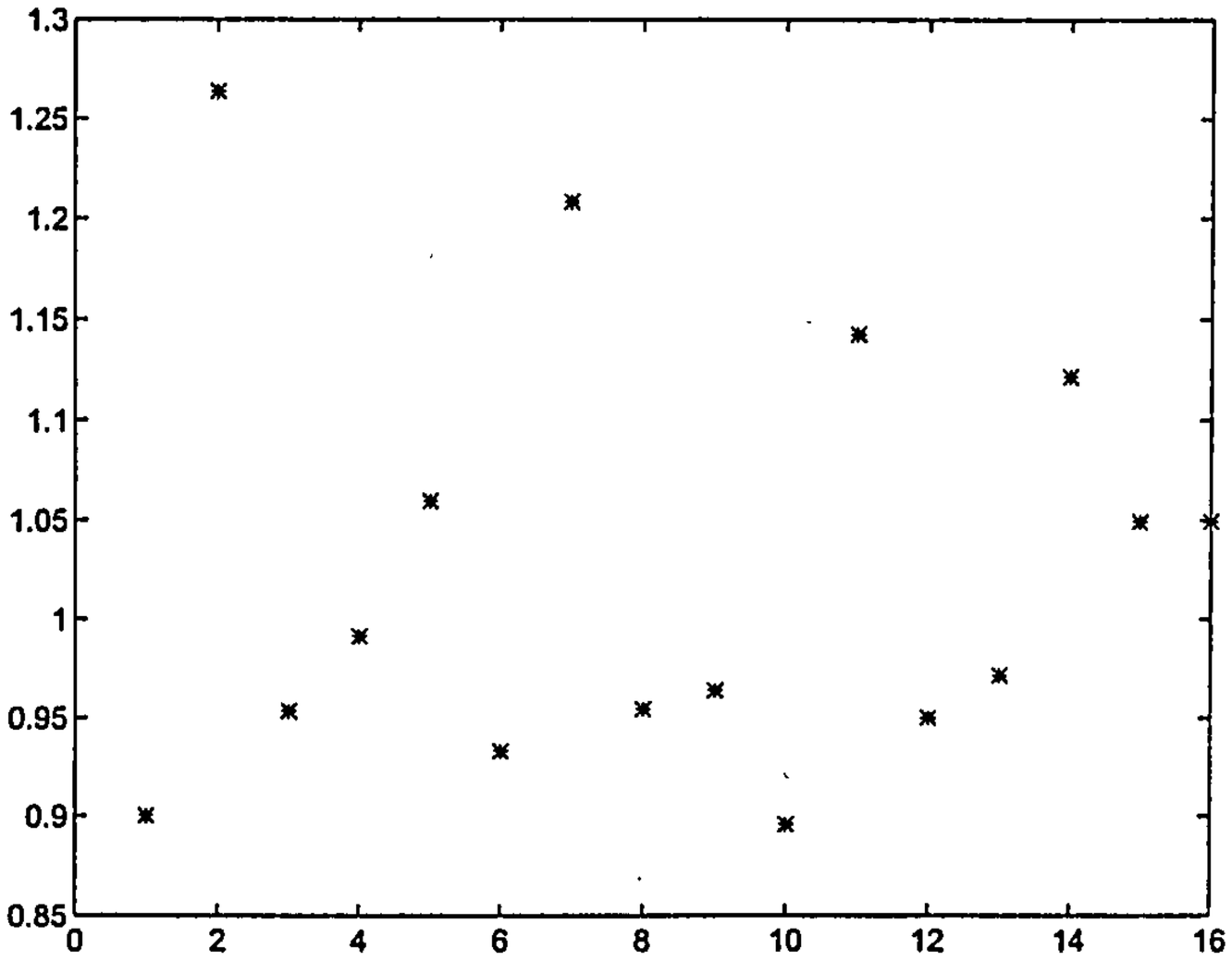


Figure 2.6: The 10 fuzzy rules overlap

2.4.3 Some results based on the analysis of variance

To illustrate the use of this method, we use the data set described in section (10.7) and an RBF network with the same number of hidden units as data points, also as described in section (10.7). In this case, because of the one to one correspondence between the hidden units and the data points, model selection and data reduction is equivalent, in order to select the best possible data points or hidden units, we need to compute:

1. all the possible values of the squared multiple correlation coefficient R^2
2. the F statistic, from which we can test the hypothesis that all the regression coefficients are zero,
3. the probability or confidence value associated with F, and
4. the 95% confidence intervals of the data points (which correspond to hidden units in this example) about the residuals.

We found that R^2 is 0.81 indicating that the model accounts for over 81% of the variability in the outputs, Y. The F statistic had a value of approximately 12 and its p–value was 0.0005. It is therefore highly unlikely that all the regression coefficients are zero.

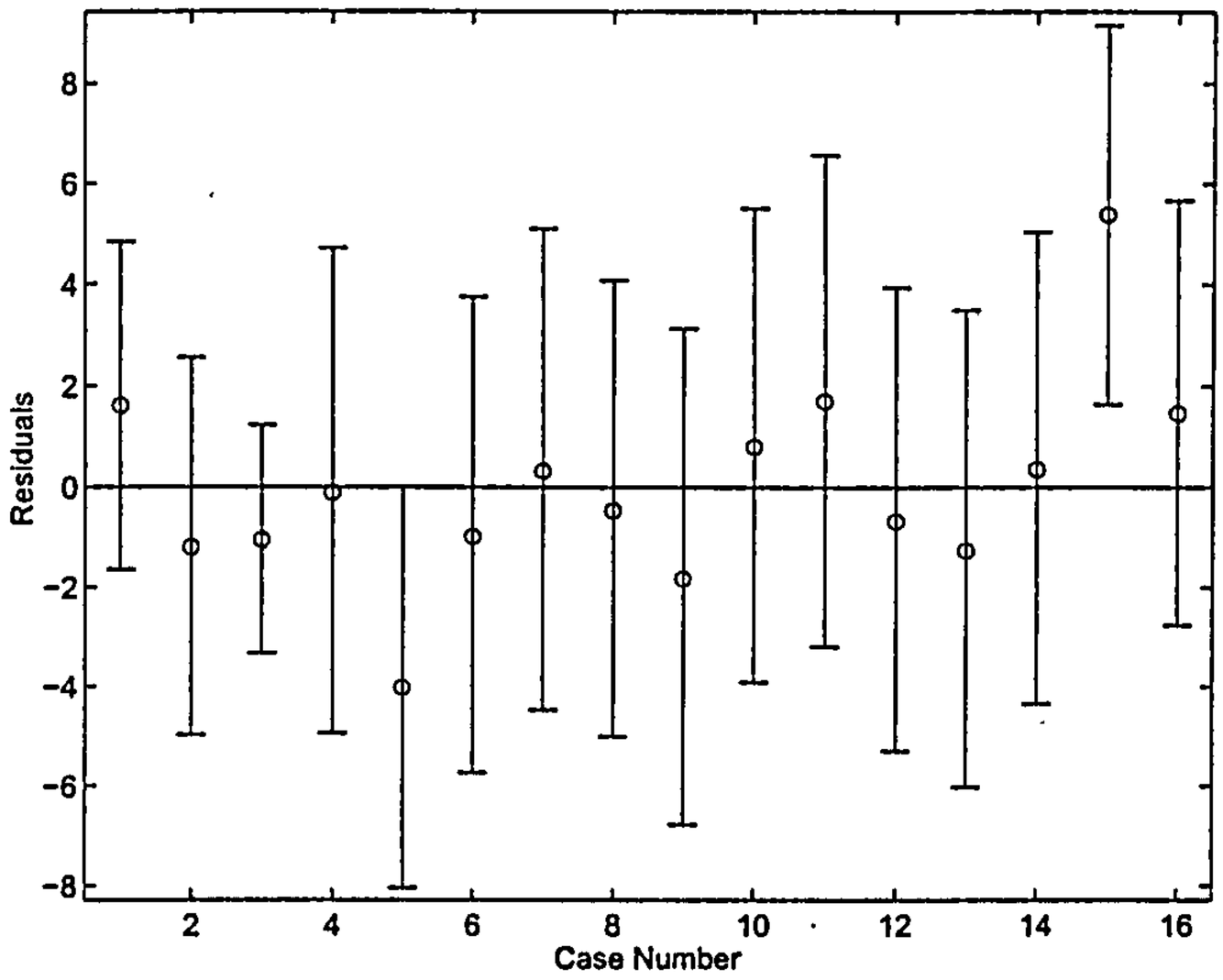


Figure 2.7: Data points selection

Figure (2.7) shows the plot of the residuals and their 95% confidence intervals for each of the sixteen data points. It can be seen that the 15^{th} and 5^{th} data points are outliers since their error bars do not cross the zero reference line.

2.5 Orthogonal optimisation algorithms

Problem : The primary motivation for the work leading to this section concerns learning algorithms for the solution of the following non–linear programming problem which arises from neural networks and fuzzy logic systems.

P1:

$$\min f(x), x \in R^n$$

Definition: A set $S \in R^n$ is said to be **convex** if and only if $x_1, x_2 \in S$ imply that $\lambda x_1 + (1-\lambda)x_2 \in S, \forall \lambda \in [0,1]$

Geometrically, this definition entails that any line segment between two points in S is always in S. Without additional generality, one can extend this definition to employ more than two points in the set S, i.e., S is a convex set if and only if $x_1, \ldots, x_k \in S$ imply that $\sum_{i=1}^{k} \lambda_i x_i \in S$ for all $\lambda_i > 0, i = 1, 2, \ldots, k, \sum_{i=1}^{k} \lambda_i = 1$.

Definition: A function f is said to be convex function if and only if for every two

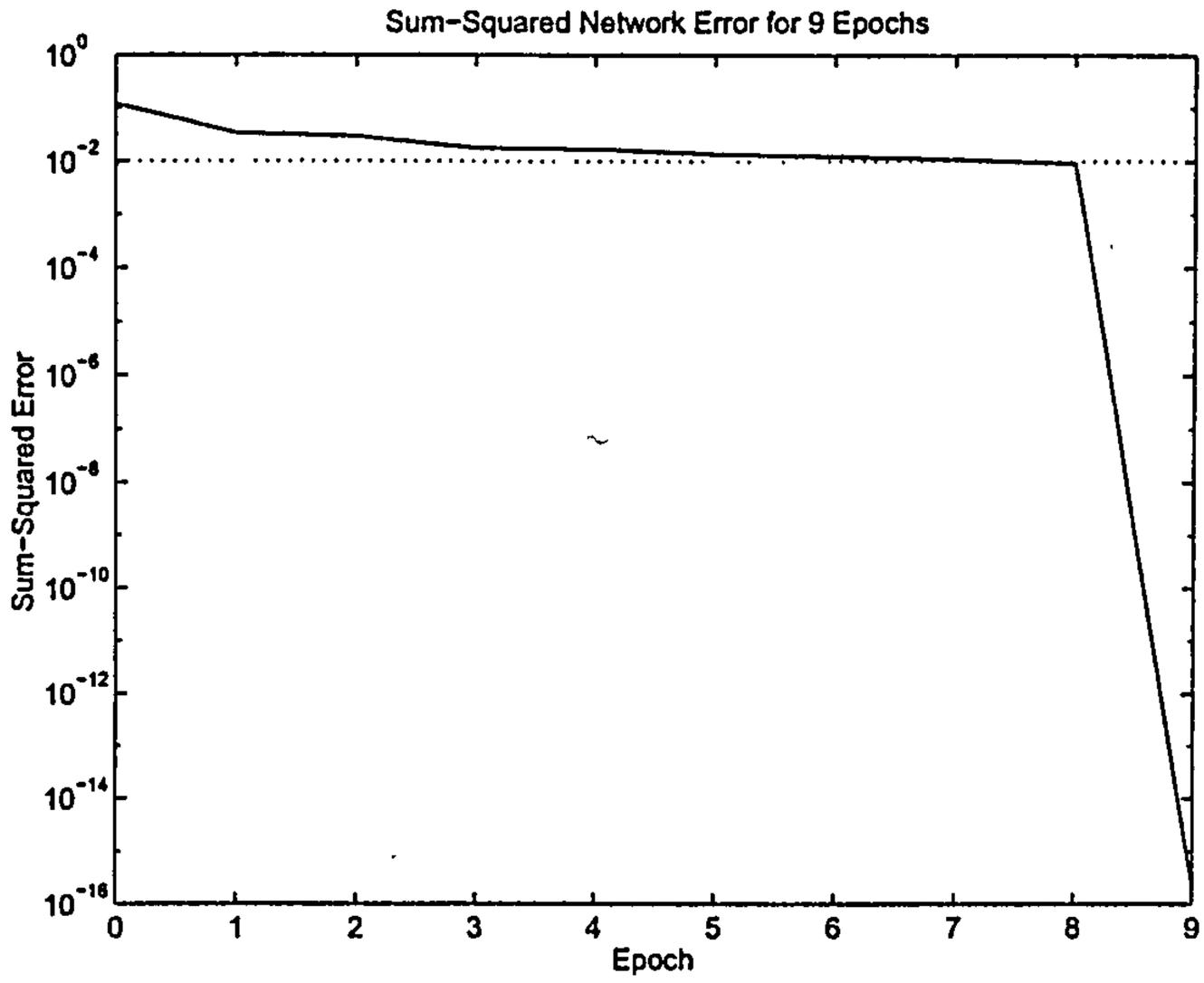


Figure 2.8: $\beta = 120$: converges with 6 neurons

elements $x_1, x_2 \in S$ and every $\lambda, 0 \leq \lambda \leq 1$, the following inequality holds:

$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2)$$

The optimization problem **P1** is non–linear in x on the feasible set R^n. We assume here that the non–linear function has at least one local minimum. If f is convex then any local minimum is a global minimum.

For a point $\ddot{x}$ to be a local minimum implies that thé first partial derivative of f, denoted by $\bigtriangledown f(x)$ is zero at $\ddot{x}$. In addition, second order necessary conditions are that $\bigtriangledown f(\ddot{x}) = 0$ and the second partial derivatives or the so-called **Hessian** matrix of f(x), denoted by $\bigtriangledown^2 f(x)$ is a **positive semi-definite symmetric** at $\ddot{x}$.

The central problem here is that the neuro-fuzzy function f is non–linear and also non-convex. This makes problem **P1** very difficult, because there are several local minima. The motivation for its solution relies on a class of search direction algorithms (SDA's).

These algorithms have a common property that during the learning process a search direction s_k is chosen such that the objective function f decreases in this direction, i.e.,

$$f(x_k + w_k s_k) < f(x_k) \qquad (2.5.1)$$

A common framework for the analysis and development of SDA's rest on a

standard quadratic function $v : R^n \rightarrow R$

P2:

$$v(x) = a + b^T x + \frac{1}{2} x^T C x \tag{2.5.2}$$

1. v(x) is the objective function,
2. a and b are constants,
3. C is positive symmetric matrix, i.e., $C^T = C > 0$.

It is very difficult to derive systematic notions that give the relative merits of SDA's for problem **P1**. However, such notions can be easily obtained for **P2** and by using the Taylor expansion of f we can say something

about the different SDA's.

To set the stage to this end, consider the series expansion of f at x_k .We can make the following second order approximation of f

$$f(x_k + w_k s_k) \doteq f(x_k) + w_k \bigtriangledown f^T(x_k) s_k + \frac{1}{2}(w_k s_k)^T \bigtriangledown^2 f(x_k)(w_k s_k) \tag{2.5.3}$$

For sufficient small step length w_k it transpires that

$$f(x_k + w_k s_k) \doteq f(x_k) + w_k \bigtriangledown f^T s_k \tag{2.5.4}$$

Taking the back-propagation algorithm for **P1** , the search direction in this case is the steepest-descent

$$s_k = - \bigtriangledown f(x_k)$$

which reveals that it is indeed a descent direction since

$$f(x_k + w_k s_k) \doteq f(x_k) - w_k \parallel \bigtriangledown f \parallel^2 \tag{2.5.5}$$

implying that $f(x_k + w_k s_k) < f(x_k)$

Extending the steepest descent direction s_k to the quadratic we find that the sufficient condition for it to be a descent direction is that

$$\bigtriangledown f^T H_k \bigtriangledown f > 0, \text{for} \bigtriangledown f(x_k) \neq 0 \tag{2.5.6}$$

where $H_k \in R^{n\times n}$ is the inverse Hessian matrix $\nabla^2 f(x_k)$.

By putting $H_k = I$, making the Hessian to be an identity matrix, the steepest descent directions are descent directions, so long as the gradient of $f(x)$ is non–zero. The identity matrix used here is a very poor approximation to the inverse Hessian. Although the method is widely used in neural networks and fuzzy system, it tends to render the learning process to be dependent on the initial conditions, the step length (called the line search) minimization and on the nature of the function. Thus, making it to be oscillatory on some problems. This is expected since f is non–convex, possessing several local minima. These are among the reasons that make the convergence of this method to be only linear.

To overcome this problem a natural attempt is to employ a search direction which explores the information embedded in the second term of Taylor series expansion above. We proceed here, firstly and without loss of generality, by setting the step length to unity we observe that the necessary condition s_k is a global minimize of the resulting quadratic is that

$$\nabla^2 f(x_k)s_k + \nabla f(x_k) = 0 \qquad (2.5.7)$$

Thus, the optimal search direction becomes

$$s_k = -H_k \nabla f(x_k) \qquad (2.5.8)$$

for some positive definite inverse Hessian matrix. The feasibility of s_k as a descent direction is obvious and this requires the positivity of H_k. This method arises from a broad class of algorithm called Newton algorithms. Newton class of algorithms attempts to approximate $\nabla^2 f(x_k)$ with H_k^{-1} so that

$$\lim_{n\to\infty} H_k^{-1} = \nabla^2 f(x_k) \qquad (2.5.9)$$

The Newton method ensures that the inverse Hessian which is obtained enables the search direction to be a descent direction, i.e. $\nabla f(x_k)^T s_k < 0$.

This leads to a linear or super–linear convergence.

2.5.1 Conjugate Gradient algorithms

Definition: A collection of search direction vectors $s_1, \ldots, s_k \in R^n$ are said to be

mutually C–conjugate with respect to a given positive definite matrix $C \in R^{n\times n}$ if for all i and j with $i \neq j$ we have $s_i^T C s_j = 0$. It is obvious that if this property holds then the set of search directions are all linearly independent.

Now consider **P2**. Along the search direction s_j from the point x_j the optimal step length w_j is

$$w_j = -\frac{s_j^T \bigtriangledown v(x_j)}{\bigtriangledown v(x_j)^T C \bigtriangledown v(x_j)} \tag{2.5.10}$$

Given a set of search directions $s_0, s_1, \ldots, s_j$, generated from a conjugate gradient algorithm for **P2**, we can consider a linear subspace spanned by these vectors as

$$L_j = \left\{ s : s = \sum_{i=0}^{j} \lambda_i s_i, \lambda_i \in R \right\} \tag{2.5.11}$$

Suppose the learning process starts from point x_0 then after j iteration steps a learning vector $\hat{x_j}$ will be in the affine manifold

$$M_j = \{x : x_0 + L_j\} \tag{2.5.12}$$

i.e. $\hat{x_j} \in M_j$. In addition, if the following orthogonality condition:

$$\bigtriangledown v(\hat{x_j})^T s_i = 0, i = 0, 1, \ldots, j \tag{2.5.13}$$

holds then $\hat{x_j}$ is a global minimize on the affine manifold. Here we present a conjugate gradient algorithm that has these properties.

A1

1. Stop if $\| \bigtriangledown v(x_j \| < \epsilon$ for small ϵ
2. Choose the step length $w_j \in R$ to minimise $v(x_j + w_j s_j)$
3. Set $x_{j+1} = x_j + w_j s_j$
4. Compute the next search direction
   $s_{j+1} = - \bigtriangledown v(x_{j+1}) + \beta_j s_j$ where β_j is given as

$$\beta_j = \frac{\bigtriangledown v(x_{j+1})^T (\bigtriangledown v(x_{j+1}) - \bigtriangledown v(x_j))}{\| \bigtriangledown v(x_j) \|^2}$$

5. Set $j = j + 1$ and go to 1.

The application of CG to our problem is equivalent to the solution of a linear equation

$$A\theta = Y. \tag{2.5.14}$$

The idea is that since we have shown that we can use the CG to solve **P2** we can use it to solve the equivalent system of equation (2.5.14). To see this, consider

$$v(\theta) = (\theta^T A A^T \theta)/2 - Y^T\theta + Y^T Y$$

and the minimum which occur when the gradient of $V(\theta)$ is zero, producing (2.5.14). This algorithm provides a powerful method for solving linear system positive definite matrix $C = A^T A$ and it follows quite closely the algorithm described for quadratic problem **P2.** Thus the algorithm takes the form

A2

1. $j = 0 : \theta, g_j = Y, \mu_j = Y^T Y, s_j = -g_j,$
2. Stop if $\| \bigtriangledown v(\theta_j) \| < \epsilon$ for small ϵ
3. Choose the step length w_j and search direction s_j :

$$q_j = As_j, r_j = s_j^T q_j, w_j\mu_j/r_j.$$

4. Set $\theta_{j+1} = \theta_j + w_j s_j, g_{j+1} = g_j + w_j q_j,$
5. Compute the next search direction

$$t_j g_{j+1}^T q_j, \beta_j = t_j/r_j.$$

$$s_{j+1} = -g_{j+1} + \beta_j s_j, \mu_j = \beta_j \mu_j.$$

6. Set $j = j + 1$ and go to 1.

Notice that unlike in the neural networks problem **P1** which require the approximation of the Hessian matrix and the gradient g and the search direction are obtained directly without the need for defining separate codes for them.

An alternative algorithm is preferable known as *modified Gram–Schmidt* (MGS). It has the same structure as above but with a modified scheme to improve convergence. We have experimented with the modified Gram–Schmidt (Grace 1993, Seber and Wild 1989) and the references quoted. In fact by choosing μ we

can stop the process at the subspace that is more representative of the system and leaving out the rest of the spaces that are not. This is a kind of model reduction.

2.5.2 Some results on optimisation

We used the orthogonal method explained with the structure of the basis functions chosen as $n_y = 0, n_u = 1, n_e = 0$ so that the process model described in section (1.5.1) corresponds to our illustrative example. We do not give details here, except to say very briefly that this orthogonal least squares model selection scheme relies on the transformation of the data matrix A into an orthogonal matrix representing a system of uncoupled, statistically independent equations so that a model selection procedure may be developed which acts on each equation individually to determine whether it is significant or not.

To have a picture of how the orthogonal least squares works in terms of complexity reduction, the same problem is solved here as in the ANOVA method.

In addition, to compare the algorithms in terms of speed, we have also used back–propagation, momentum (faster backpropogation) and Newton–Raphson (Levenberg–Marquardt) algorithms. We used a personal PC running Matlab with a few hundreds kilobyte of cache. We implement a backpropogation based in steepest descent using the same illustrative example. To implement the gradient descent we create a neural network with 10 hidden layer neurons and learning rate 0.01. In this technique the parameters of the neural network, such as the weights and the biases, are moved in opposite direction to the error gradient.

The Newton–Raphson algorithm introduced in (Davis and Vinter 1984) is more powerful than the backpropogation algorithm, but requires more memory. The learning rule is update with

$$\triangle\theta = (J^T J + \mu I) J^T \hat{\epsilon} \tag{2.5.15}$$

where J is the Jacobian matrix of the derivatives of the estimate of the error to each parameter θ and μ is some small number.. We set sum of squares error goal to 0.01, meaning that if the sum of squares error reaches 0.01 then the

optimisation should terminate. For the RBF, β is set to 1.

MODEL STRUCTURE	TIME	EPOCH	FLOPS
BACKPROPOGATION	25.6	11276	67105794
FASTER BACKPROP.	6.2	2333	14003772
LEVERBERG–MARQUARDT	0.3	7	2100584
BASIS FUNCTION	0.2	4	21692

(2.5.16)

The results are shown on the table[8] Results for a faster backpropogation is also used (*faster* refers to the fact that the optimisation algorithm uses a combination of the gradient and the change of gradient of the error function to update the next search direction).

The backpropogation optimisation could not solve the problem for 10000 epochs, so we extended the maximum number of epochs until the error goal was achieve after 11276 epochs. However, the radial basis function networks using Billings algorithm only required 4 neurons to converge. The algorithm works by creating a neuron at each epoch using the 16 data points. At each iteration the algorithm picks one input vector out of the 16, the data point that minimises the error the most, to create a hidden neuron. The error goal is then monitored and the process continues until the error goal is met. We obtained the result indicated earlier in section (10.7) that four were required as may be seen from evolution of the mean square error depicted in figure (2.9). Therefore, the RBF provides us with the choice of choosing a system that can solve problem with fewer neurons, although Newton–Raphson can be used if enough computational memory is available. As shown, the back propagation is too slow for real industrial applications.

2.5.3 Comparing different RBFs

The problem consists of input variable P and one target variable T with data prepared from truncated version of electricity data. Three different RBF networks with different membership functions are trained in two stages. First, a Gaussian mixture model is trained using the EM algorithm and the centres of this model are used to set the centres of the RBFs. Second, the output weights (and biases) are determined using the conjugate gradient algorithm of the quadratic forms.

[8]Epochs are the number of iterations of the optimisation algorithms for neural networks and refer to the number of neurons for the RBF using the Arnordi MGS algorithm. Times are measured in seconds.

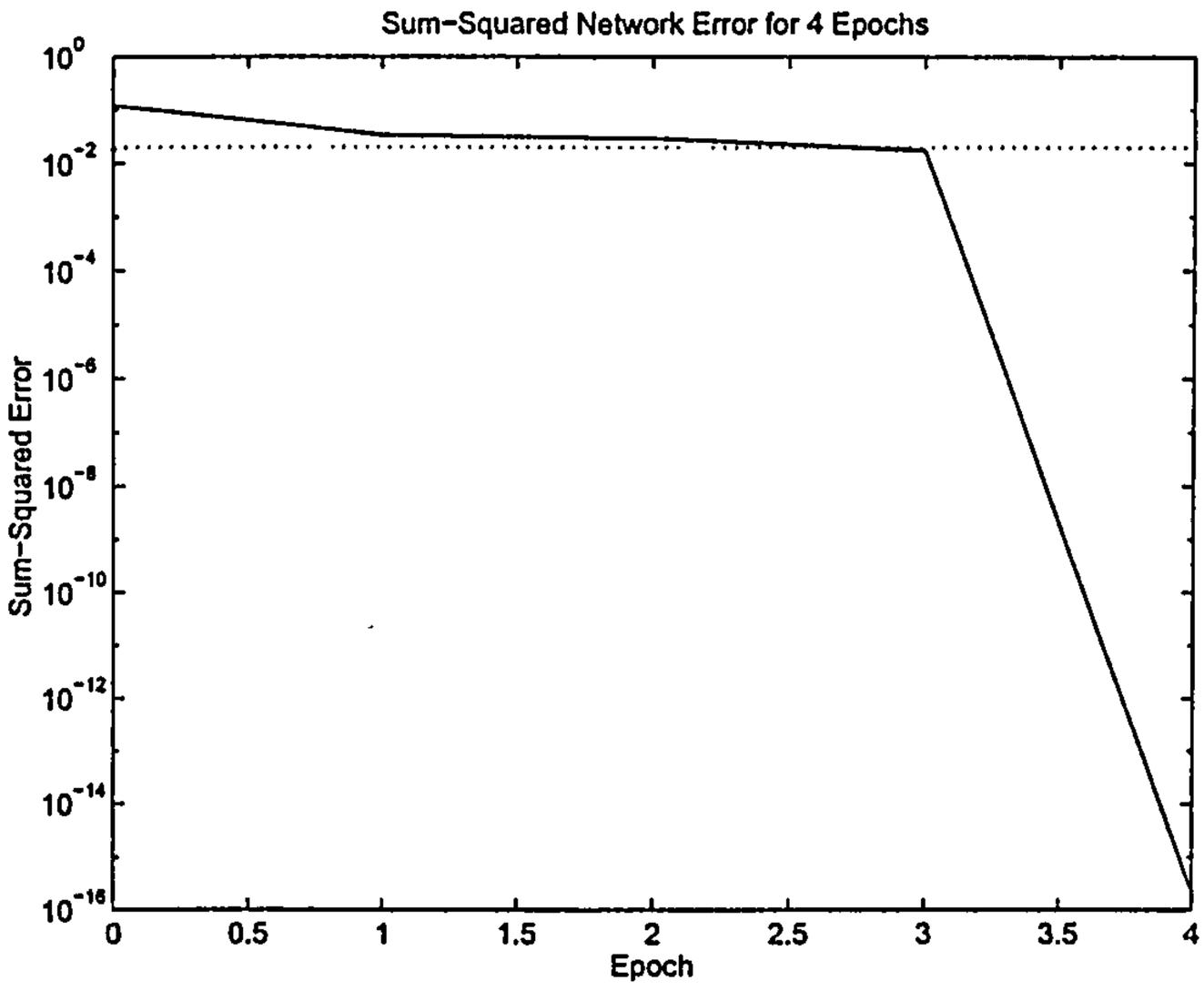


Figure 2.9: Evolution of mean square with Arnoldi algorithm.

In the first stage of the construction, we initialize a radial basis function network with a suitable set of centres. The centres are determined by fitting a Gaussian mixture using the EM algorithm. In fact, other fuzzy classification algorithms can be used (Mallison and Bentley 1999).One mixture model is initialized using a small number of iterations of the K-means algorithm.

The next phase of the experiment we generates a matrix A of the hidden unit activations where each row corresponds to one pattern. These hidden unit activation's represent the design matrix for the RBF. The matrix is formed from the transfer functions acting on the squared distances between each basis function centre and each pattern in which each row corresponds to a data point. With the design matrix A and the output target T, we are now equipped for optimisation with the error quadratic form using any of the conjugate gradient methods. In figure (2.10) there are three graphs, the plot of the data point of the product truncated electricity demand over 2000 minutes , the approximation using Gaussian membership functions, Thin plate spline and $r^4 \log r$ RBFs, where r is the distance of each data point from the centre of membership functions.

Application of the conjugate gradient algorithm to general quadratic programming problems was considered in (Tankeh, Azvine and Mamdani 1997a, Tankeh and Mamdani 1997b). With the truncated version of electricity data the RBFs produce

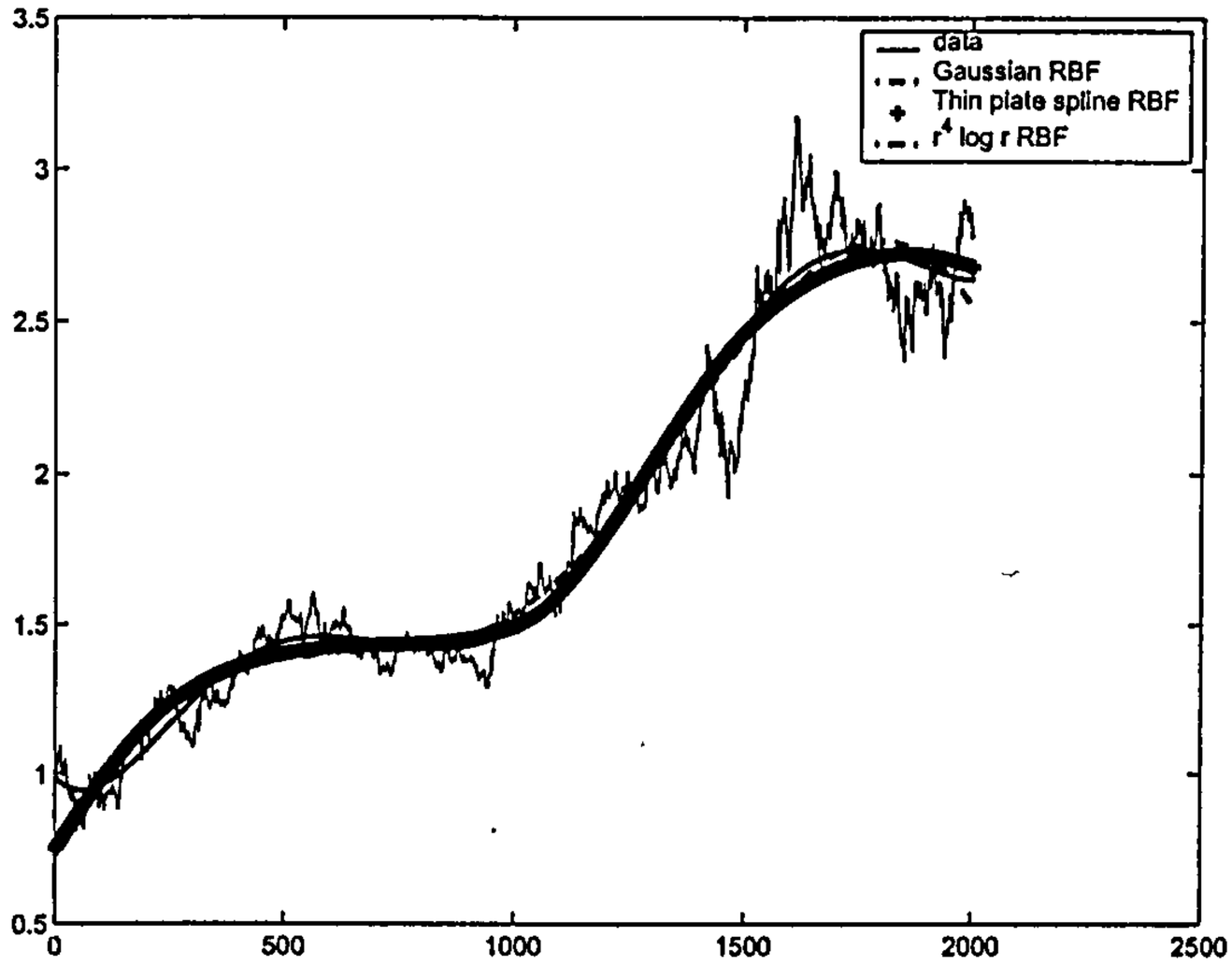


Figure 2.10: Different membership functions for RBFs

the following design matrix A, with 2001 entries. Only few entries are shown here.

$$1.0e+014 * \left\{ \begin{array}{lll} 0.2974 & 0.8507 & 0.0006 \\ 0.2964 & 0.8484 & 0.0006 \\ 0.2954 & 0.8462 & 0.0006 \\ 0.2944 & 0.8439 & 0.0006 \\ 0.2934 & 0.8417 & 0.0006. \end{array} \right\}.$$

$y_{k+1} = f(y_k, y_{k-1})$ is the output, (y_k, y_{k-1}) is the input and f is the radial basis function. Also note that we use

$$y_{k+1} = \theta_1\phi_1 + \theta_2\phi_2 + \epsilon_k,$$

where ϕ s are the basis functions and θs are to be determined.

Using the conjugate gradient algorithm the optimisation converges iteration 6, with solution $1.0e-012 * (-0.0524 - 0.12740.1277 - 0.45570.08170.0000)$ in one instance of the experiment.

2.5.4 Results for truncated electricity data

Using the method of regression, we compute the estimation together with $100(1-\alpha)$ confidence intervals. From the R–statistics (Draper and Smith 1998), we found the model explains 99.8% of the variability in the data. Also, for model

reduction, trying to detect data points that were not contributing much in RBFs approximation, we constructed the residuals, which is the different between observation and the predicted values. Unfortunately, the residuals are correlated. Therefore, we compute a standardised forms of the residuals, as explained in the earlier, producing a student t–distribution, which from the theory of analysis of variance are uncorrelated.

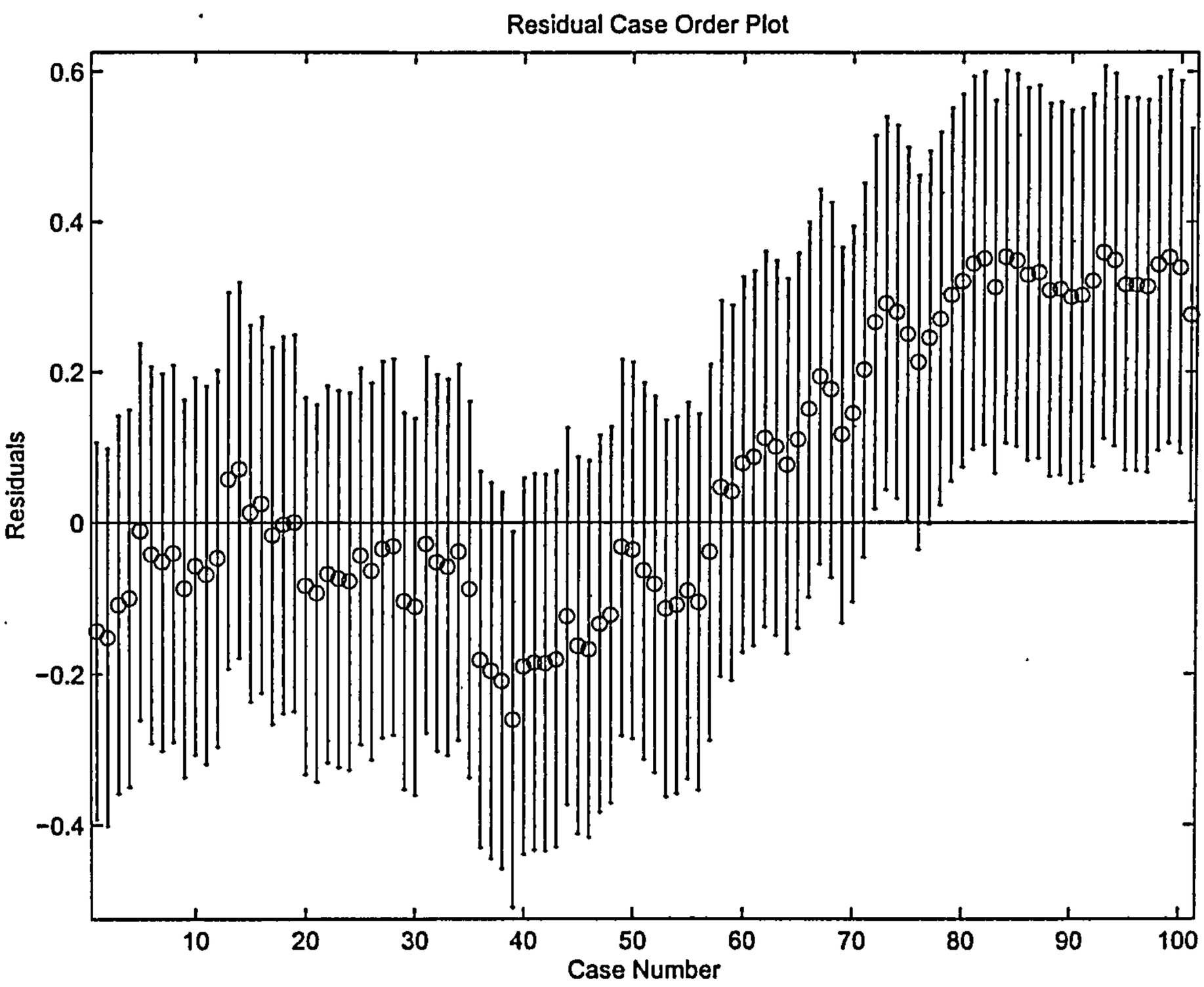


Figure 2.11: Model reduction with residuals

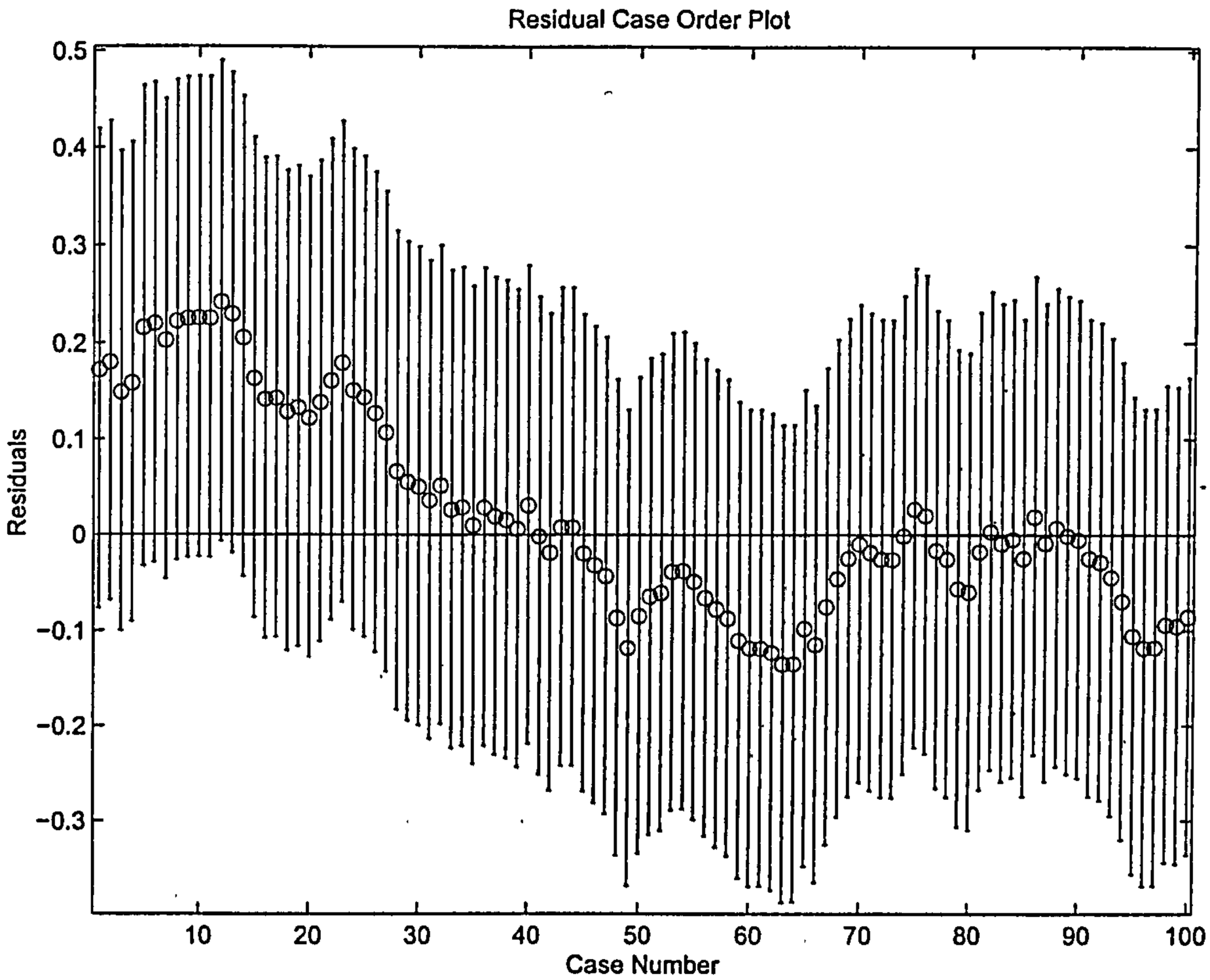


Figure 2.12: Model reduction with residuals

2.5.5 Regression with daily claims data

The training data are arranged in the manner that has been explained, choosing a second order Markov scheme.

$$1.0e+006 * \left\{ \begin{array}{lll} y_{k+1} & y_k & y_{k-1} \\ 2.4087 & 8.3550 & .3550 \\ 1.6408 & 2.4087 & 8.3550 \\ 4.5810 & 1.885 & 2.4087 \\ 0.9906 & 4.5810 & 1.6885 \\ 0.6463 & 0.9906 & 4.5810 \\ 0.4625 & 0.6463 & 0.9906 \\ 0.4576 & 0.4625 & 0.6463 \\ 0.9095 & 0.4576 & 0.4625 \\ 0.6406 & 0.9095 & 0.4576 \\ 0.9580 & 0.6406 & 0.9095. \end{array} \right\}$$

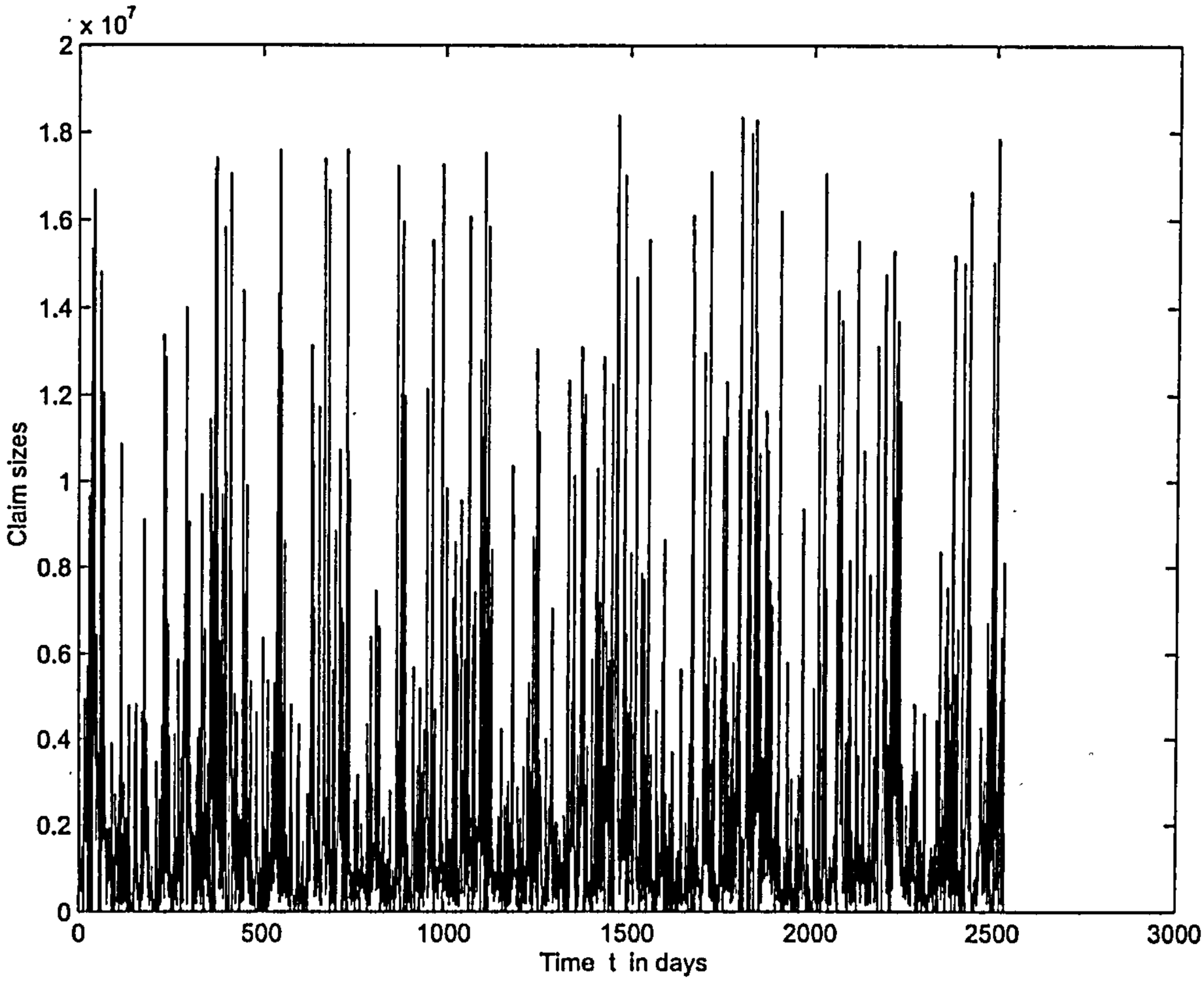


Figure 2.13: Daily claims data

The least–squares regression estimates 0.36200.2571. as the parameters., with confidence interval

$$0.3244 \quad 0.3997$$
$$0.2194 \quad 0.2947.$$

using the estimate we then compute the residuals, the standardised student t–distribution. The model reduction graph is shown in figure (2.15).

2.6 Model reduction with constraint optimisation

2.6.1 Regularisation and Ridge regression

We shall illustrate this idea using truncated electricity data. When computing the regression problem, it is possible to allow the constrain solution θ to lie in the ball

$$\theta^T\theta < \mu^2,$$

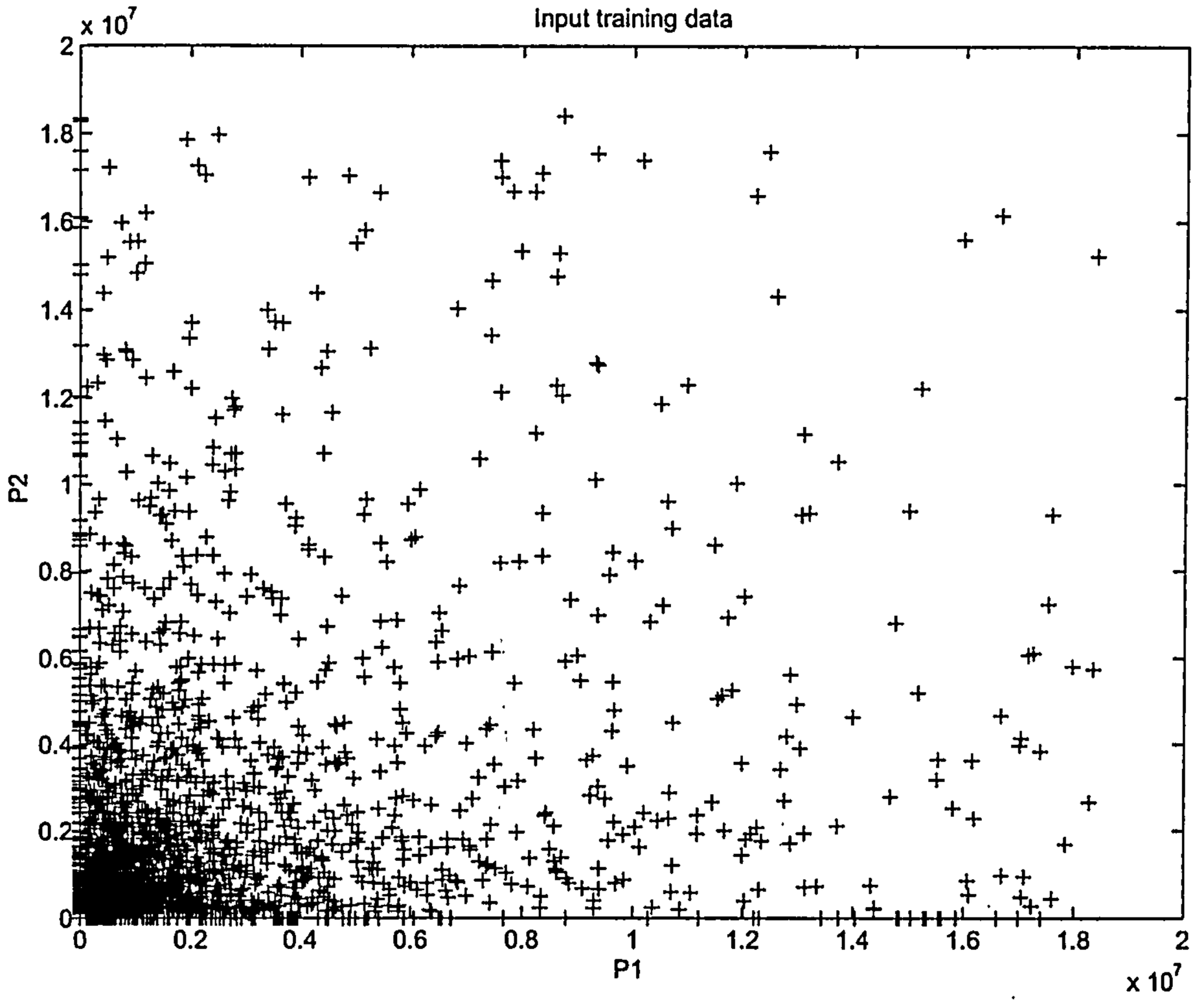


Figure 2.14: training data: claims data

where μ is the radius, called the *regularisation* radius. This states a maximum (Euclidean norm) value for estimate θ and avoids getting solution of very large norm. The important question that arise is whether the solution minimises the residual as the least–square estimator does and whether it is biased too. By considering the quadratic form

$$S(\theta) = (Y - A\theta)^T(Y - A\theta) + \mu(\theta^T\theta - R^2), \tag{2.6.1}$$

where μ is the Lagrange multiplier, and equating the derivative equal to zero, we obtain

$$(A^TA + \mu I)\theta = A^TY,$$

where μ is a function of the R. As shown on the figure(2.16), the estimate θ changes with μ. It can be shown that for increasing μ, the bias of θ increases, but the variance falls. Again when the design matrix A is poorly condition, having very large condition number, the drop in variance more than it compensates for the bias.

To appreciate this assertion, it is important to look again at the information

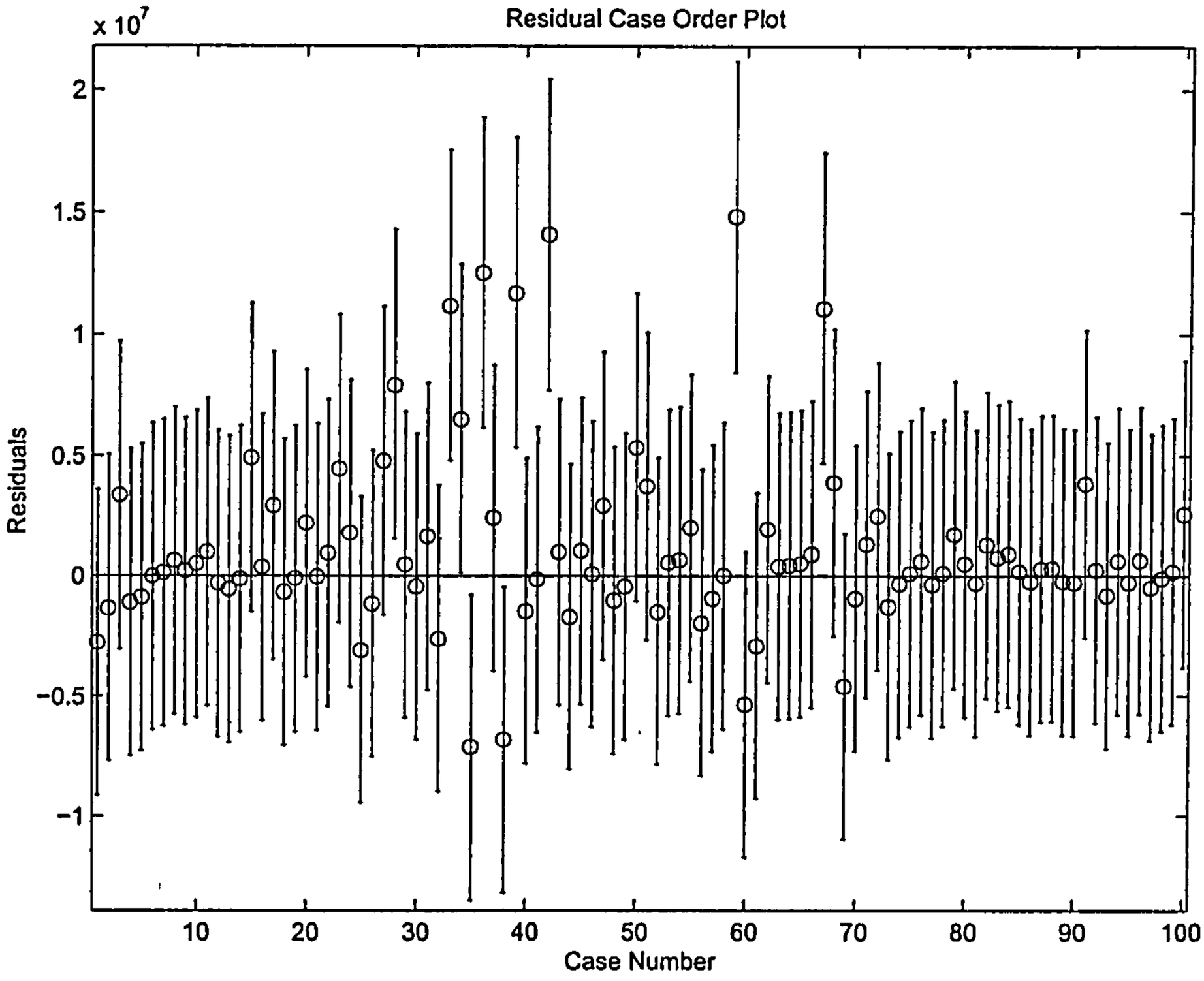


Figure 2.15: Model reduction plot: Daily claims data

matrix A^TA. Let θ_l be the least–square estimate of θ. However, if the information matrix is ill–conditioned, having very large spread of eigen values with some closer to zero then the estimate $\hat{\theta}$ will be very far from θ_l. Infact, if $L^2 = \|\hat{\theta} - \theta_l\|^2$ then the variance of error of estimate from least square is

$$E\left[L^2\right] = \sigma^2 trace(A^TA)^{-1} = \sigma^2 \sum_{i=1}^{p} \frac{1}{\lambda_i}.$$

We see here that the matrix A, which enters our training data plays a major part in controlling the type of estimates that we obtain. The matrix $(A^TA)^{-1}$ is called *a design* matrix and we shall see in chapter (4) how to obtain it via the method of convex optimisation of covariance structure.

$$E\left[L^2\right] \geq \sigma^2 \frac{1}{\lambda_{\min}}.$$

A large value of $E\left[L^2\right]$ means that at least one component of θ has a large variance, which implies that the smallest eigen value of the design matrix (A^TA) is very small. This is an ordinary least squares estimate
$\theta_l = (0.1717, -0.0272, -0.01770.0304)$ with norm

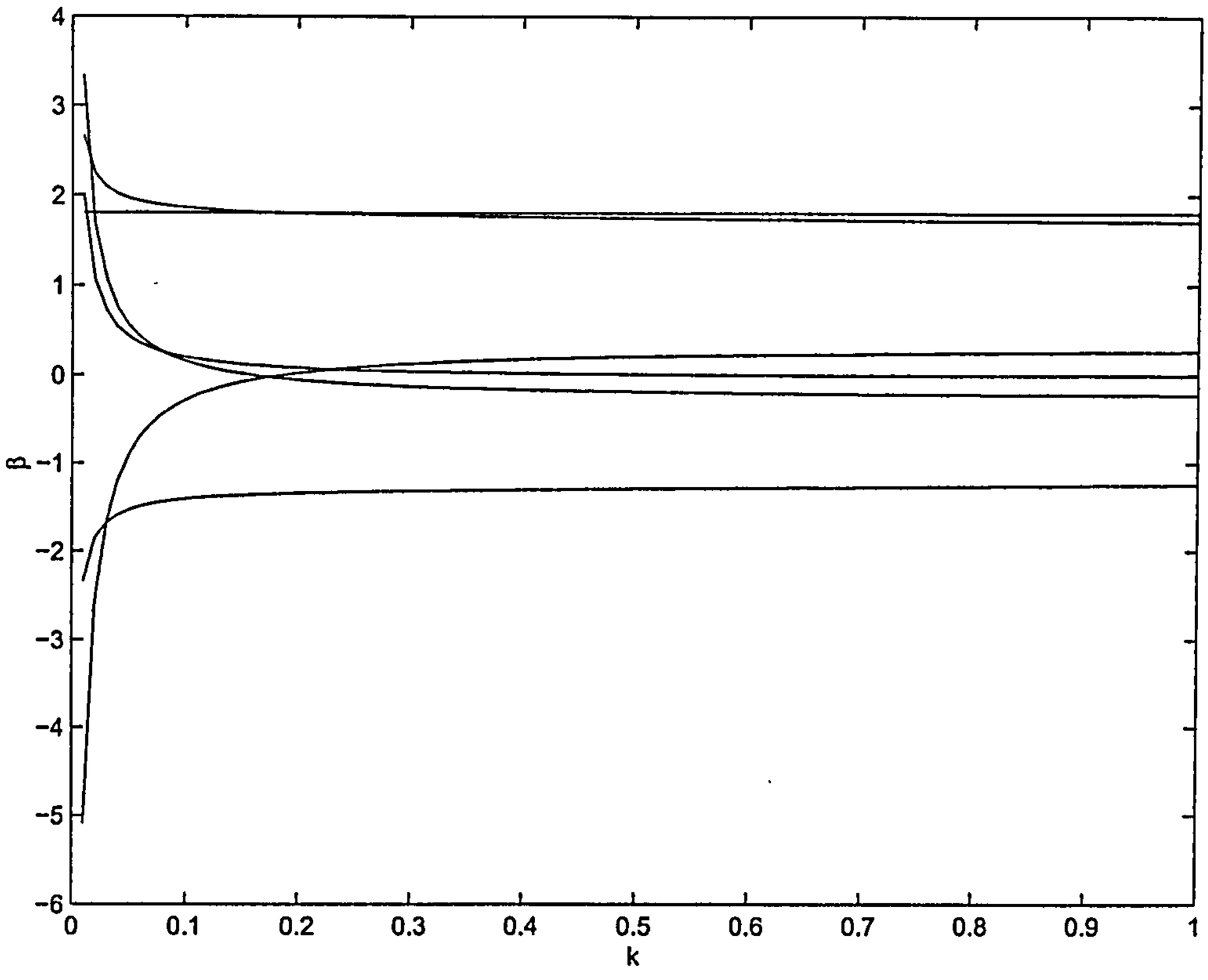


Figure 2.16: Effects of ridge parameters

$0.1774, \hat{\theta} = (0.0033, -0.0005, -0.0004, 0.0006)$ with norm 0.0034. For a proper choice of μ, the ridge regression is the estimator among all the unbiased estimators that minimises the residuals sum of squares (Zacks 1981) and the mean square error

$$E\left[L^2\right] = \sigma^2 \sum_{i=1}^{p} \frac{\lambda_i}{(\lambda_i + \mu)^2} + \mu^2 \sum_{i=1}^{p} \frac{\gamma_i}{(\lambda_i + \mu)^2},$$

where $\gamma_i = Q\theta$, where if the matrix of eigen vectors. Also the orthogonal matrix

$$Q = \begin{bmatrix} -0.3608 & -0.6151 & -0.6113 & -0.3431 \\ -0.6139 & -0.3490 & 0.3770 & 0.5994 \\ -0.6066 & 0.3666 & 0.3369 & -0.6198 \\ -0.3536 & 0.6045 & -0.6088 & 0.3728 \end{bmatrix}.$$

diagonalises the information matrix $A^T A$. If we collect all the unknown parameters in a vector Θ then we can formulate the sample maximum likelihood condition on the n data points. Infact θ_l is

$$(X^T X)^{-1} X^T Y,$$

which is the maximum likelihood estimate of

$$L(\theta) = \frac{1}{\sigma^n} \exp\left(-(1/\sigma^2)\left\{(Y - A\theta)^T(Y - A\theta)\right\}\right) \tag{2.6.2}$$

and it also minimises the quadratic form (2.6.1) irrespective of the variance of the error σ^2. We can also differentiate $L(\theta)$ with respect to σ and equating equal zero to obtain

$$\hat{\sigma}^2 = \frac{1}{n} Y^T P Y$$

with a better unbiased estimate

$$\hat{\sigma}^2 = \frac{1}{n-p} Y^T P Y, \tag{2.6.3}$$

where P is the idempodent matrix with property $P^2 = P$ and p is the number of parameters. It has rank 4. Hence, $\hat{\sigma}^2 = 1.0857e-005$. Therefore, we can use the method (Zacks 1981) to obtain the value of μ As

$$\mu = \frac{p\hat{\sigma}^2}{\|\theta_l\|^2} = 2.4603e-004$$

where θ_l is the least square estimate. The above two methods of regularisation is very different from those normally used in RBFs (Zhu and Billings 1996). It can be shown that mean square error (MSE) for the ridge regression is

$$\sigma^2 \sum_{i=1}^{p} \frac{\lambda_i}{(\lambda_i + \mu)^2} + \mu^2 \sum_{i=1}^{p} \frac{\gamma_i}{(\lambda_i + \mu)^2}.$$

The variation of μ depends on the eigen values of the information matrix. For our example, figure (2.17) shows that it increases between zero and one. We experimented with this optimisation and got $\mu = 0.7$.

2.6.2 Application of F–Distribution

To proceed, let us review the development leading up to this point. In (Davis and Vinter 1984) the model reduction problem is formulated as a F–statistic tests, a framework that was just broad and did not proceed to any specific application. Referring the reader to some history, the F–statistical test is analogous to Pearson likelihood ratio test which goes as follows: If H_0 is the null hypothesis, representing the parameter set Θ_0 that is a subset of a much larger set Θ. Then

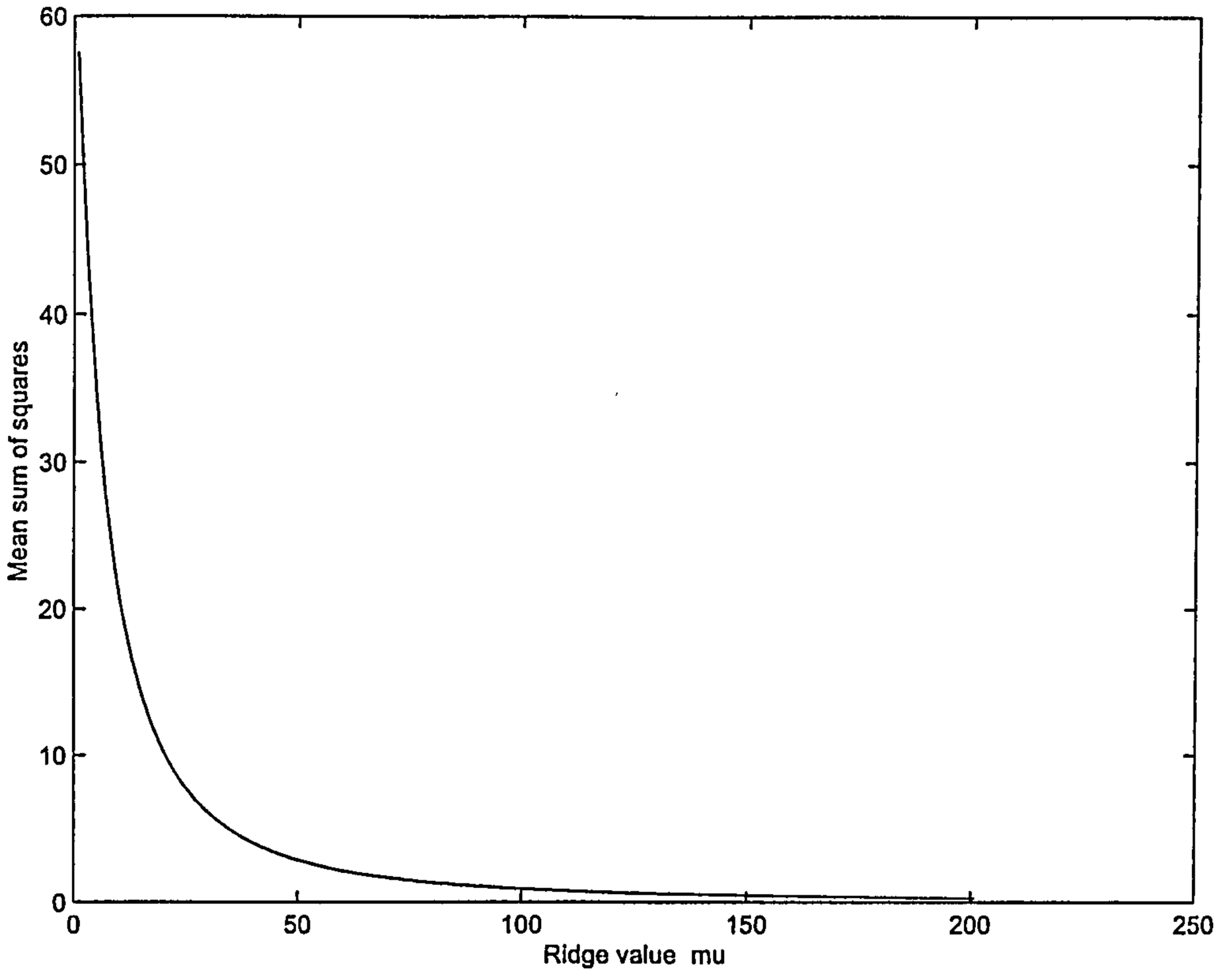


Figure 2.17: Effects of ridge regression

ratio test is defined as

$$R = \frac{\max_{\theta \in \Theta_0} L(\theta)}{\max_{\theta \in \Theta} L(\theta)},$$

It is obvious that $0 \leq R \leq 1$. The hypothesis is accepted if $R \leq C_\alpha$, where C_α is a function of a small give number α.

Consider the vector $\hat{\theta}$ that minimizes the RBFs error $\|Y - A\theta\|$. Thus, the following quadratic forms are established

$$\begin{aligned} \|Y\|^2 &= \|Y - A\hat{\theta}\|^2 &+ \|A\hat{\theta}\|^2 \\ &= Y^T(I-P)Y &+ Y^TPY \\ &= \text{square of Residual} & \text{square of estimated output} \\ &= \hat{\epsilon} & \hat{Y}, \end{aligned}$$

$P = A(A^TA)^{-1}A$ is a *projector operator* that projects any vector into the range of A. Therefore, P is idempotent (Allwright 1988). Returning to the model reduction problem with the hypothesis

$$\theta_{r+1} = \theta_{r+2} = \ldots = \theta_p = 0$$

and

$$\theta_1, \theta_2, \ldots, \theta_r$$

are arbitrary. Let $L = [O_r, I_{p-r}]$ be a $(p-r) \times p$ matrix , which is partitioned into a zero matrix O_r of order $(p-r) \times r$ and an identity matrix of order $(p-r) \times (p-r)$. Using the procedure set out in the ridge regression, we need an optimisation of the

$$S(\theta) = (Y - A\theta)^T (Y - A\theta) + \theta^T L\Lambda, \tag{2.6.4}$$

where Λ is the Lagrangian vector of dimension $(p-r) \times 1$. The normal equations are

$$\begin{aligned} -2A^T(Y - A\theta) + L^T\Lambda &= 0 && (2.6.5) \\ L\theta &= 0. && (2.6.6) \end{aligned}$$

Let $\hat{\theta}$ be the least square estimate with no model reduction constraint. Then from the first equation, we obtain the constrained least squares estimate $\acute{\theta}$, given by

$$\acute{\theta} = (A^T A)^{-1} \left[A^T Y - \frac{1}{2} L^T \Lambda \right] = \hat{\theta} - \frac{1}{2} (A^T A)^{-1} L^T \Lambda.$$

From which we get

$$\Lambda = 2 \left[L(A^T A)^{-1} L^T \right]^{-1} L\hat{\theta}.$$

Hence,

$$\acute{\theta} = \left[I - C^{-1} L^T (KCL^T)^{-1} L \right] \hat{\theta},$$

where $C = A^T A$ is the usual Fisher information matrix.

The estimation of the variance is obtained using similar ideas leading to equation (2.6.3), with respect of the quadratic form and after simplification

$$\acute{\sigma}^2 = \hat{\sigma}^2 + \frac{1}{n} \hat{\theta}^T L^T B^{-1} L\hat{\theta}, \tag{2.6.7}$$

where $\hat{\sigma}^2$ is the variance from the non–constraint problem. Through out we have seen that the residual s can be written as

$$Q = Y^T P Y$$

where P is an idempotent matrix. An important relation between the distribution of Y and Q is that (Grabill 1976), if $Y \sim N(\mu_y, S)$ and SP is idempotent

$(i.e.(SP)^2 = SP)$ of rank p then

$$Q \sim \chi(p : \lambda), \ \ \lambda = \frac{1}{2}\mu_y^T P \mu_y$$

with degree of centrality λ.

In addition, we know that an F distribution arises when we are comparing two random variables of quadratic forms. In this case two variances–one with constrained and the other without constraint. This is the application of the idea set out in (Davis and Vinter 1984) about the use of F–distribution.

Therefore, the model reduction testing of hypothesis reduces the ratio of the two quadratics:

$$F = \frac{n-p}{p-r}\frac{Y^T P_1 Y}{Y^T P_2 Y},$$

the P_1 is the idempotent matrix that relates the estimate of the variance σ^2 under the constraint and P_2 is the estimate without the constraint.

It turns out that:

$$P_1 = A(A^T A)^{-1} L^T * (L(A^T A)^{-1} L^T * L(A^T A)^{-1} A^T,$$

which is idempotent and of rank $p - r$.

$$Y^T P_1 Y \sim \text{ chi–squares distribution with parameter } (p-r; \lambda_1)$$

where $p-r$ is the rank of the idempotent matrix P_1 and the parameter of centrality

$$\lambda_1 = \frac{1}{2\sigma^2}\theta^T L^T (L(A^T A)^{-1} L^T L\theta,$$

where $L\theta = (\theta_{r+1}, \theta_{r+2}, \ldots, \theta_p)$ which is equal to zero if the null hypothesis is true. According to (Grabill 1976) two quadratic forms are independent if their corresponding idempotent matrices are orthogonal and we can verify that

$$P_1 P_2 = 0.$$

Hence, the F statistics above are distributed like a central F distributions with the degree of freedoms $p - r$ and $n - p$.

2.6.3 Confident interval for constraint least–squares

We have discussed how to develop a Model reduction scheme using F statistics but what about confidence intervals. The procedure for computing confidence intervals for such a scheme follows a similar procedure as that of parameter estimation. We need to compute the variance of the parameter estimates and from these we can deduce the confidence interval. We leave out technical details and just present the general procedure and results.

2.6.4 Experiments with constrained least squares

We can summarise the Model reduction algorithm developed from the F–distribution of (Davis and Vinter 1984) as follows:

1. Set up the training data as a design matrix A and the output Y.
2. Now choose a design matrix and the value of r such that $\theta_{r+1} = \theta_{r+2} = \ldots = \theta_p = 0$ and the corresponding L.
3. Compute the constraint least squares and find the value of the optimiser $\acute{\theta}$.
   Using the truncted electricity data we have

$$\acute{\theta} = (0.1718, -0.0303, -0.00000.0000).$$

4. Compute the root mean squares error (RMSE).
5. Compute the R^2– the amount of the response variability explained by the model.
6. Compute the F–the overal F statistics for the model reduction.
7. Compute P–the associated significance probability.
8. Compute the upper and lower confidence interval of the estimate $\acute{\theta}$.

We conducted the following experiments:

- Using the truncated version of electricity data $r = 2$. For thetruncated version of electricity data we have $p = 4$ and choose $r = 2$

$$L = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Parameter	Estimate	Lower confidence	Upper confidence
$\hat{\theta}_1$	0.1707	−0.008609	0.3501
$\hat{\theta}_2$	−0.0134	−0.2109	0.1485
$\hat{\theta}_3$	0		
$\hat{\theta}_4$	0		
RMSE	R^2	F	P
0.01898	0.02834	2.872	0.05893.

- Using the truncated version of electricity data $r = 1$. For thetruncated version of electricity data we have $p = 4$ and choose $r = 1$

$$L = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Parameter	Estimate	Lower confidence	Upper confidence
$\hat{\theta}_1$	0.1667	−0.01092	0.342
$\hat{\theta}_2$	0		
$\hat{\theta}_3$	0		
$\hat{\theta}_4$	0		
RMSE	R^2	F	P
0.01894	0.02739	5.577	0.01917.

- Using the truncated version of electricity data $r = 3$. For the truncated version of electricity data we have $p = 4$ and choose $r = 3$

$$L = \begin{bmatrix} 0 & 1 & 0 & 1 \end{bmatrix}$$

Parameter	Estimate	Lower confidence	Upper confidence
$\hat{\theta}_1$	0.1703	−0.000961	0.3502
$\hat{\theta}_2$	−0.02884	−0.2118	0.1541
$\hat{\theta}_3$	−0.0134	−0.1942	0.1674
$\hat{\theta}_4$	0		
RMSE	R^2	F	P
0.01903	0.02851	1.917	0.1281.

With no model reduction the $P = 0.2105$ which means that, with probability of 21.05%, we are better off doing model reduction, as opposed to a $P = 0.05893$ which has a very small probability for $r = 2$. Therefore, the best model order for this system is 2.

2.7 Summary

In the work leading to this thesis, we have found neural networks the multi-layer perceptron networks(MLPs) to be problematic in a number of ways, not the least of which is the poor convergence of network weights under training due to the inherent nonlinear nature of the weight updating algorithms. We have paid attention has to radial basis function networks (RBFs) which have much better learning properties. This is mainly because of the straightforward linear relationship between the network weights and the output response. RBF networks therefore offer considerable promise in terms of network stability and robustness proofs, largely because the network can easily be defined by means of an equation set.

As is the case with modelling methods in general, and particularly with neural networks, a problem occurs in terms of determining a suitable RBF network structure.The key problem is deciding both the quantity and positions of basis function centres. The choice directly affects the goodness of fit of the overall network approximation achieved. There should be sufficient number of basis functions to span the entire input domain, but not too much. We have seen that the use of a large number of basis functions obviously results in computational problem, giving rise to the what is known as over–parameterisation. This makes the network in capable of good generalisation.

This chapter also shows the importance of model reduction and clustering to the modelling of RBFs within the general framework of Markov time series. The model reduction algorithms presented uses modern methods of analysis and optimisation. All the necessary methods have been developed from subject areas that have been neglected by RBFs researchers.

With model reduction, it is not just about stepping foot in water now and then, getting feet wet and then leaving. This chapter presents the first step towards a unified treatment of model complexity in nonlinear regression that can be approximated with a radial basis functions. We have used results from statistical analysis of variance (ANOVA) to introduce a set of model selection principles that can be used to reduce the number of hidden neurons in RBFs, therefore, reducing the complexity of model.

Furthermore, the chapter concludes by presenting a new approach of model reduction for RBFs that is based on constraint optimisation of quadratic cost function. This new approach also provides methods for evaluating the efficiency of

the basis functions procedures.

Chapter 3

Modelling Correlation and Transient effects

3.1 Introduction

In chapter (2) we made the following assumptions about our *compound renewal process:* (1) for large number of customer sources, the demand process is asmptotically Gaussian, (2) the noise sequences impacting the discrete state–space representation are uncorrelated, and (3) mean and the variance of the resulting demand process are stationary.

Put in another way, the major objective of this chapter rests on three fundamental conjectures. The first is that the essential characteristics of data points arising from demand processes(including those collected from simulation of queues) are correlated. The second is that each statistical parameter of interest such as the mean or variance is time dependent. Thus, the variance at each time step is related to its predecessors in a very complicated way.

This chapter shows how to formulate a structure for modelling these additional effects in a manner that enable us to: (1) compute transient estimates of mean and variance of demand values using RBF networks; (2) estimate steady values of queue length from simulation results. Good examples are the data collected from IBM switch, the simulation of the queues of I/O devices when one computer server is communicating with another during a data communication process and simulation of queues for the sake of studying the relative merits of different queueing models.

In all these cases, we are interested in obtaining the steady state results from a

finite number of batch means $(X_1, X_2, \ldots)$ over a finite time period. In a simulation of this kind, it is impossible to run experiment to for ever (i.e. using an infinite number of batches) in order to estimate steady state. When a finite number of batches are used to estimate the distribution of some parameter of interest, the problem becomes a transient problem and the batches record are usually correlated.

The problem of identifying the steady state mean queue length from the initial transient response using simulation has been widely studied (Law and Kelton 1991) and Pawlikowski (1990). However, all existing procedures explicitly make similar assumptions that are made in RBFs research–that the observed process is in steady state(Law, 1977). Thus, using auto–regressive approach (Fishman, 1978 , Schruben, 1983, Bischak, Kelton and Pollock, 1993, Heidelberger 1992), these researchers take no account of the fact that the noise may be correlated or that the statistical properties may be time dependent.

Initial transient effects invalidate the assumption of stationary. One solution of transient effects is to use logarithmic transformation to mitigate such effects and to minimize bias due to initial transient (or time trends). Similar problems are encountered in queueing systems (Snell and Schruben 1984).

3.1.1 Steady state in queue simulation

Note that the computation methods suggested for modelling queue is tailor–made for PDP theory. Since this PDP approach were completely new, we were required to do simulation of $M/G/1/N$ to compare results. These is the set up one of the simulations: $\lambda = 0.1$, $G = E_k(\mu_0 t)$(Erlang distribution, $N = 20$, $\mu_0 = k/8$. Hence, the offer load $\rho = 0.8$. We now set the simulation run to $10,000$ and run the experiment 100 replications.

The result of the queue length against replication is shown in figure (3.1). The straight line shows the mean value and the results over all the 100 replications fluctuate about that mean.

To illustrate the concept of batching the simulation time to $30,000$ minutes, make up 301 batches with 10 observations per batch. Here are the results of the simulation

1. Initial Time Truncated : 0.

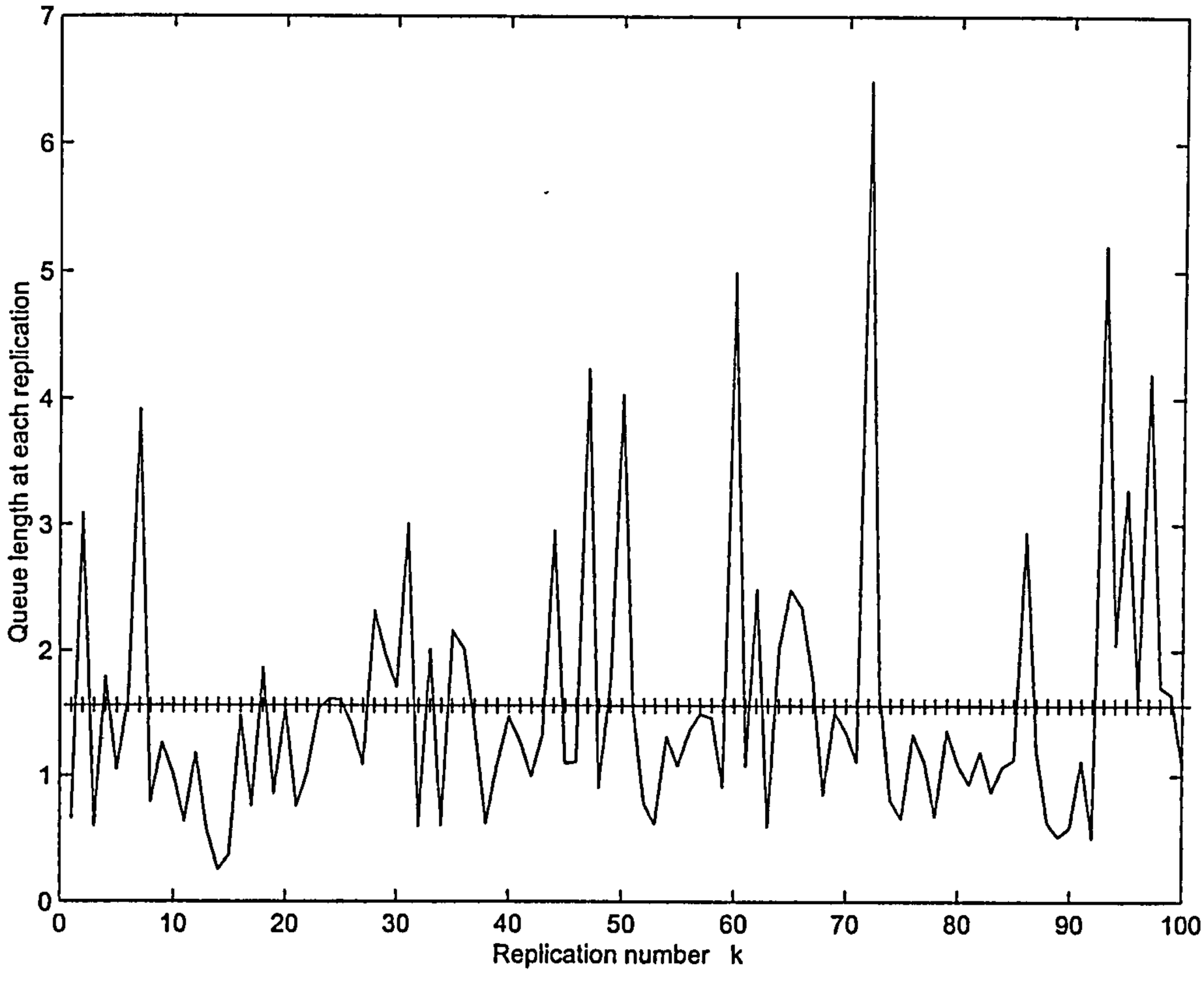


Figure 3.1: Queue lengths vs replications

2. Number of Batches : 301.

3. Number of Observations Per Batch : 10.

4. Number of Trailing Obs'ns Truncated : 3.

5. Estimate of Covariance Between Batches : 0.262.

We need to compute and display the sample ACF of the batch means, along with the upper and lower standard deviation confidence bounds, based on the assumption that all autocorrelations are zero beyond lag zero. This is shown in figure (3.4).

Figure (3.3) shows the ensemble batch average over time using 10 replications of batching. Using the average over different replications we now have sufficient information to carry out further batching across different times to estimate the mean queue length from the transient as explained above.

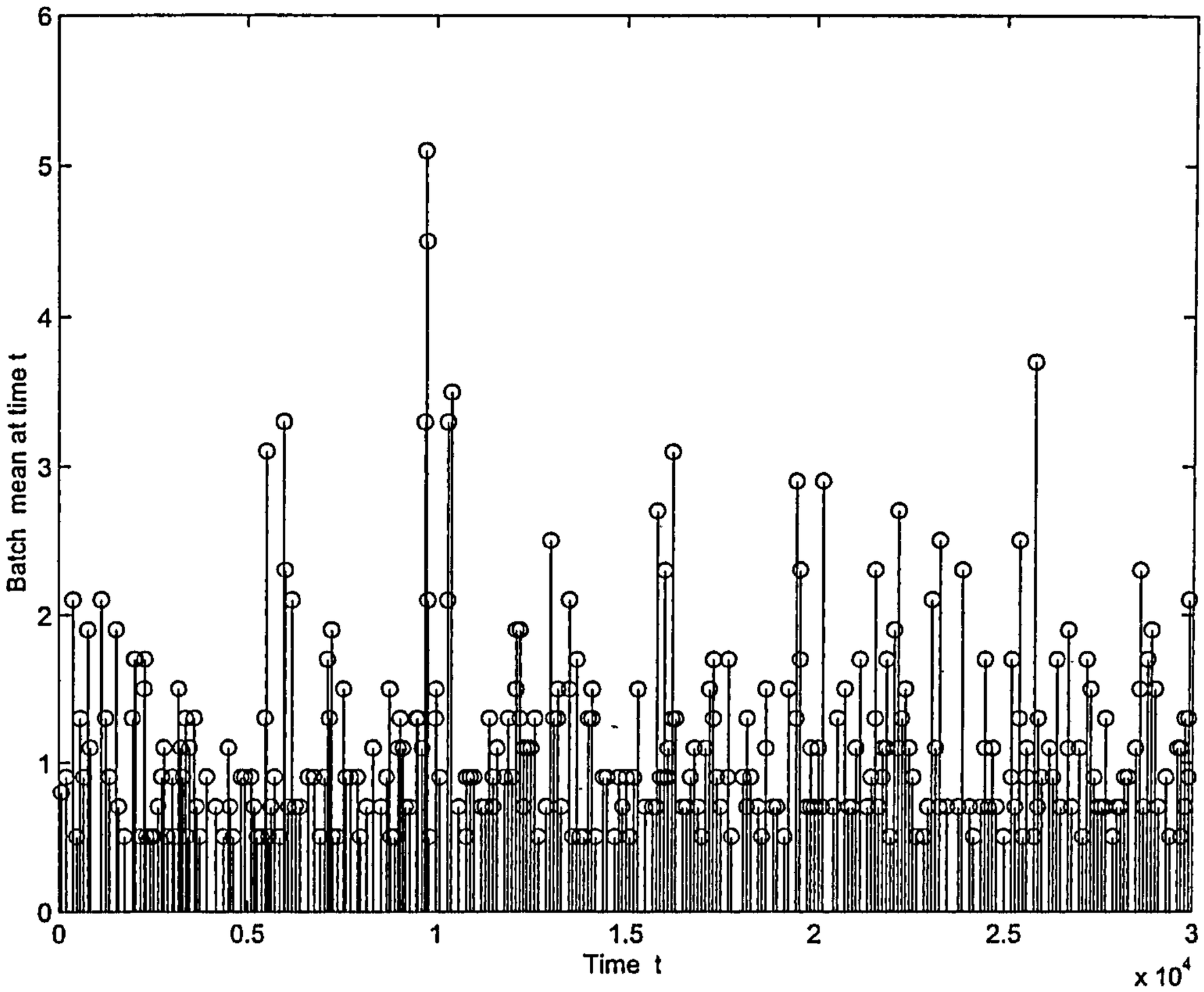


Figure 3.2: Batch mean

3.2 Correlated noise sequences

Consider the model RBFs model

$$y_t = a_t\theta + \epsilon_t. \tag{3.2.1}$$

We assume that the error ϵ_t follows a first order Markov scheme

$$\epsilon_t = \rho\epsilon_{t-1} + v_t. \tag{3.2.2}$$

Then ϵ_t is the serially correlated. We assume the following about v_t and ρ. This correlated noise structure was used in (Davis and Vinter 1984).

$$E[v_t] = 0, \ \ |\rho| < 1, \tag{3.2.3}$$

$$E[v_t v_s] = \begin{cases} \sigma^2 & \text{if} \quad t = s \\ 0 & \text{if} \quad t \neq s \end{cases} \tag{3.2.4}$$

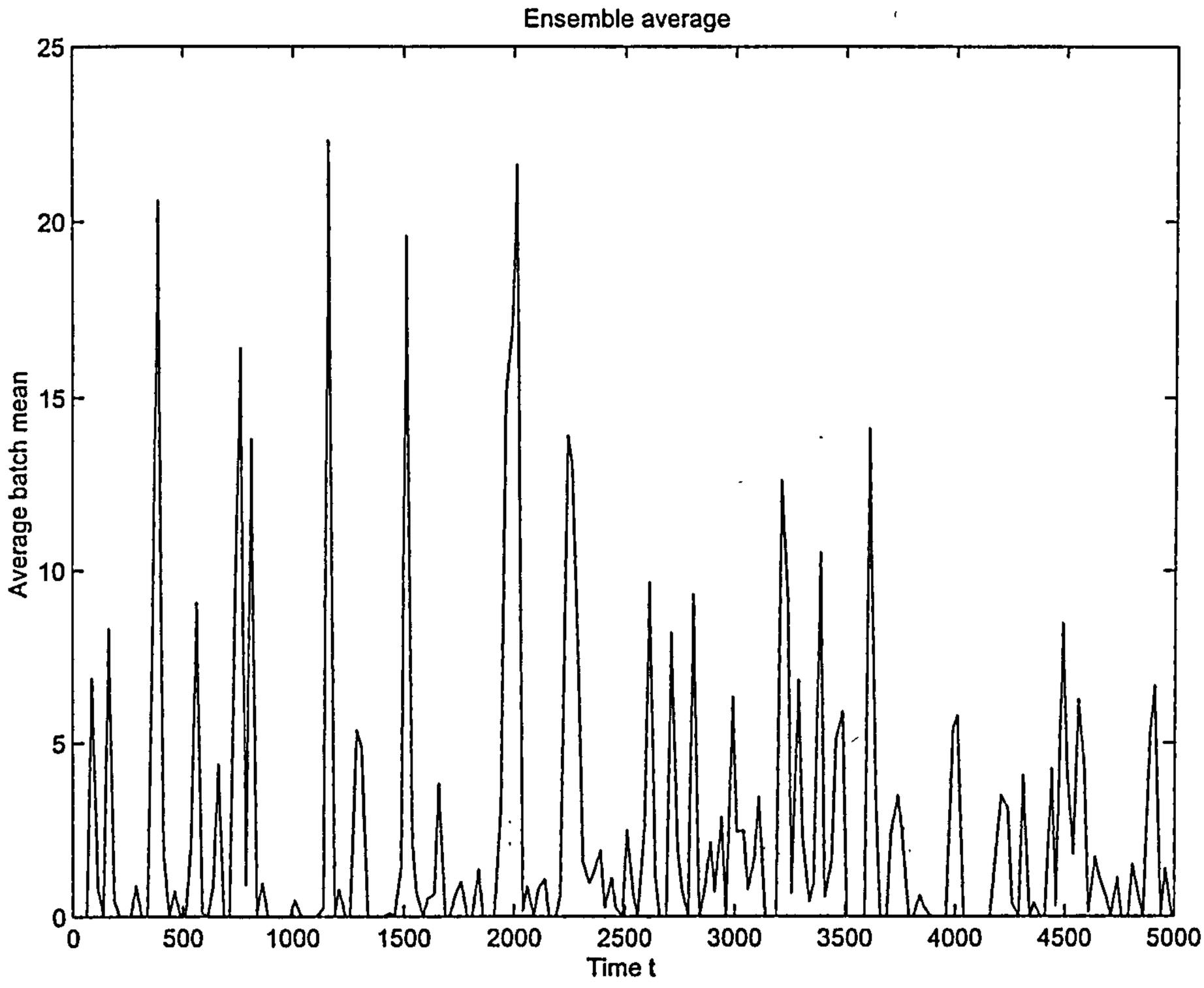


Figure 3.3: Batch average

The v_t term is known as white noise; it has constant mean and variance for all periods and is not serially correlated. Since ρ is nonzero it introduces some persistence into the system when there are shocks from the white noise term.

Taking an approach in (Davis and Vinter 1984), we can expand our expression for ϵ_t :

$$\epsilon_t = \sum_{k=0}^{\infty} \rho^k v_{t-k}. \tag{3.2.5}$$

The current disturbance is a declining average of the entire history of the white noise term. In order to say anything about the use of ordinary least squares(OLS) in the presence of autocorrelation, we need to find the mean $E[\epsilon_t]$ and variance $E[\epsilon_t^2]$ of the disturbance ϵ_t. Since the mean of the disturbance is a linear combination of random variables all with mean zero, the disturbance has a mean of zero. Hence, the error sequence $\epsilon_0, \epsilon_1, \epsilon_2$ is an L_1- *Martingale* (Davis 1993). As of the variance

$$\begin{aligned} \epsilon_t^2 &= \left(v_t + \rho v_{t-1} + \rho^2 v_{t-2} + \ldots\right)\left(v_t + \rho v_{t-1} + \rho^2 v_{t-2} + \ldots\right), \\ \epsilon_t^2 &= v_t^2 + \rho^2 v_{t-1} + \rho^4 v_{t-2}^2 + \ldots + 2\sum_{r<s}\sum \rho^r \rho^s v_{t-r} v_{t-s}. \end{aligned}$$

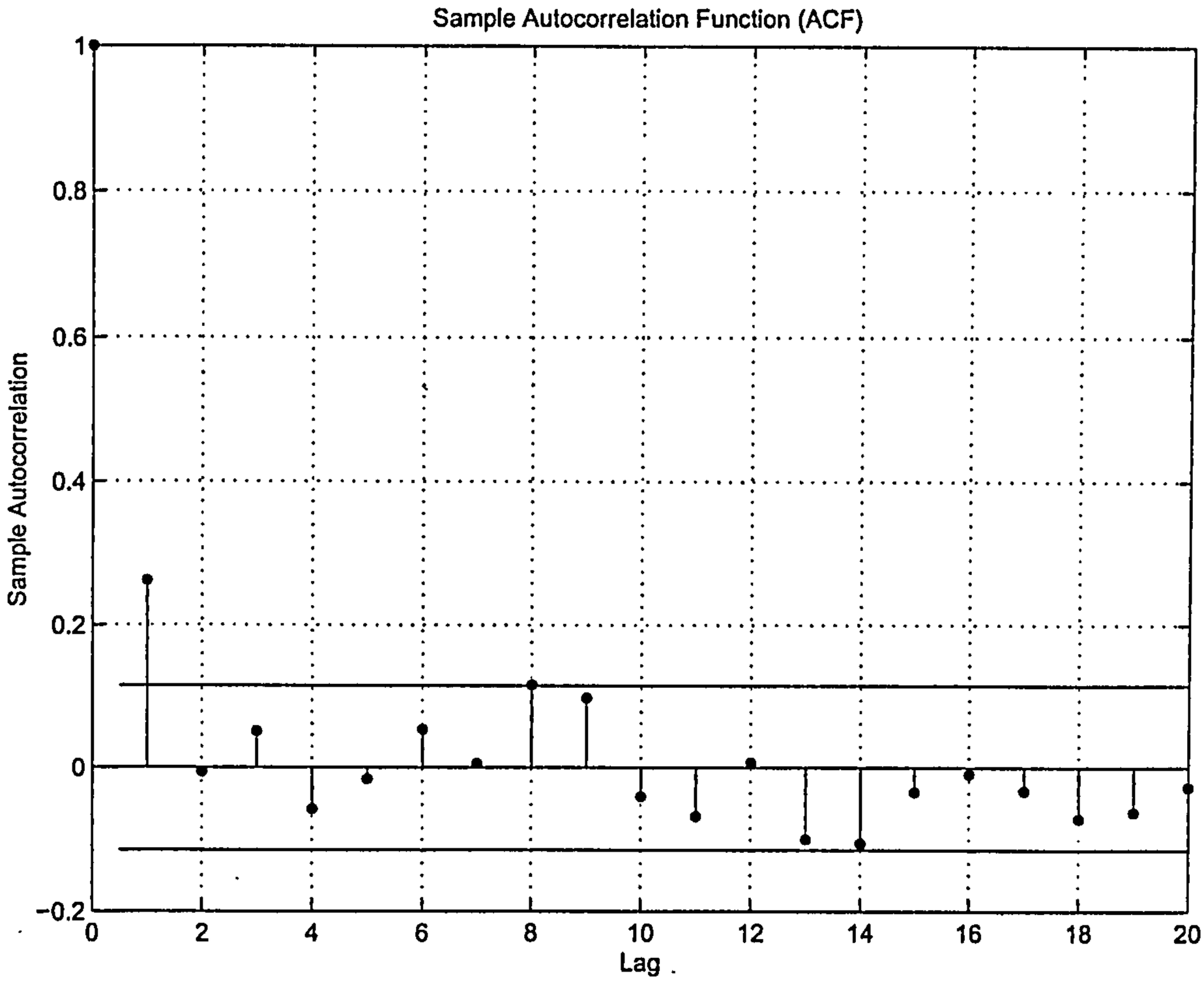


Figure 3.4: Batch average

Upon taking expectations the terms under the double sum are all cross products and drop out. That is, their time subscripts do not match and we have assumed that there is no serial correlation in the white noise term. Thus, using a bit of our human capital about infinite series,

$$E\left[\epsilon_t^2\right] = \sigma^2(1 - \rho^2 + \rho^4 + \ldots) = \sigma\left[\frac{1}{1-\rho^2}\right] = \sigma_e^2. \tag{3.2.6}$$

Applying the same tricks of the trade that we used for the expectation of ϵ_t^2 we find

$$E\left[\epsilon_t\epsilon_{t-1}\right] = \rho\left[\frac{\sigma^2}{1-\rho^2}\right] \tag{3.2.7}$$

$$E\left[\epsilon_t\epsilon_{t-2}\right] = \rho^2\left[\frac{\sigma^2}{1-\rho^2}\right] \tag{3.2.8}$$

$$\vdots \tag{3.2.9}$$

$$E\left[\epsilon_t\epsilon_{t-r}\right] = \rho^r\left[\frac{\sigma^2}{1-\rho^2}\right]. \tag{3.2.10}$$

Putting everything together in an error covariance matrix gives us

$$E\left[\epsilon\epsilon^T\right] = \sigma_e^2 \begin{bmatrix} 1 & \rho & \rho^2 & \dots & \rho^{n-1} \\ \rho & 1 & \rho & \dots & \rho^{n-2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \rho^{n-1} & \rho^{n-1} & \dots & \dots & 1 \end{bmatrix}, \tag{3.2.11}$$

where ϵ is a vector of errors and n is the number of observations. Let us consider the effects of autocorrelation on our conventional OLS estimator in RBFs. In order to throw more light on this we will consider the very simple RBF model with one hidden neuron

$$y_t = \theta a_t + \epsilon_t \tag{3.2.12}$$

with $E\left[\epsilon_t\right] = 0$ and the estimate of θ denoted as $\hat{\theta}$. Recall that any regression computation based in the standard practice of RBFs will give a variance of

$$Var\left[\hat{\theta}\right] = \sigma_e^2 \left(\sum y_t^2\right)^{-1} \tag{3.2.13}$$

in the least squares sense. Using the recipes provided in (Zacks 1980, we arrive at

$$\begin{aligned} Var\left[\hat{\theta}\right] &= \sigma_e^2 \left(\sum y_t^2\right)^{-1} \\ &* \left[1 + 2\rho \frac{\sum_2^n y_t y_{t-1}}{\sum y_t^2}\right. \\ &+ +2\rho^2 \frac{\sum_3^n y_t y_{t-2}}{\sum y_t^2} \\ &+ \left. \dots + 2\ \rho^r \frac{\sum_{r+1}^n y_t y_{t-r}}{\sum y_t^2}\right] \end{aligned}$$

which is smaller than that reported above.

Of course, this has considerable concequences on optimal design of training data, where we noted in chapter (2) the need for choosing the training data such that the error covariance of the estimator is as small as possible.

The import of this study is that you had better not have a data trend that are corrupted with correlated noise. The caveat emptor, as in just the coloured noise case, is that OLS is not efficient.

3.3 Our modelling framework

We now recall equation (1.5.3):

$$y(t) = f(y^{t-1}, u^{t-1}, \epsilon^{t-1}) + \epsilon(t) \tag{3.3.1}$$

where,

$$\epsilon(t) = [\epsilon_1(t), \ldots, \epsilon_m(t)]^T \tag{3.3.2}$$

is the noise on the outputs, $y(t)$, and y^{t-1}, u^{t-1} and ϵ^{t-1} respectively denote output, input and noise sequences as follows:

$$y^{t-1} = [y(t-1), \ldots, y(t-n_y)], \tag{3.3.3}$$

$$u^{t-1} = [u(t-1), \ldots, u(t-n_u)], \tag{3.3.4}$$

$$\epsilon^{t-1} = [\epsilon(t-1), \ldots, \epsilon(t-n_\epsilon)]. \tag{3.3.5}$$

As we have mentioned, $f(.)$ is a non-linear, m–dimensional vector function of many variables, so the input–output relationship is a non-linear mapping.

Remember that in the daily insurance claims and electricity consumption data sets, there are no inputs $u(t)$. Such systems are autoregressive. But with the rubber consumption data set , there is input $u(t)$. In any case the correlated noise model we want to study is of the form

$$y(t) = f(y^{t-1}) + \epsilon(t), \tag{3.3.6}$$

where

1. $f(y^{t-1}, \theta)$ represents the deterministic component of the current demand as a function of any information known at time $t-1$, including past errors $(\epsilon_{t-1}, \epsilon_{t-2}, \ldots,)$ and observations $(y_{t-1}, y_{t-2}, \ldots,)$ and any other parameter vector θ.

2. $\epsilon(t)$ is the random component representing the innovation in the mean of $y(t)$, this is in keeping with the idea of (Davis and Vinter 1984). Note that you can also interpret the random disturbance as the single-period–ahead forecast error.

3. For a scalar case, the variance of $y(t)$ condition on $t-1$

$$var_{t-1}(y(t)) = E_t(\epsilon_t^2) = \sigma_t^2, \tag{3.3.7}$$

where

$$\sigma_t^2 = \sigma_0^2 + \sum_{i=1}^{r} a_i \sigma_{t-i}^2 + \sum_{j=1}^{s} b_j \epsilon_{t-j}^2. \tag{3.3.8}$$

The value of writing a model in this form can be appreciated by considering a first–order autoregression

$$y_{t+1} = \theta y_t + \epsilon_{t+1} \tag{3.3.9}$$

with ϵ is i.i.d. $\sim N(0, \sigma^2)$. Future values of y for this process depend on $(y_t, y_{t-1}, \ldots,)$ only through the current value y_t. This makes it extremely simple to analyze the dynamics of the process, make predictions or evaluate the likelihood function. For example, equation (3.3.9) is easy to solve by recursive substitution,

$$y_{t+m} = \theta^m y_t + \theta^{m-1}\epsilon_{t+1} + \theta^{m-2}\epsilon_{t+2} + \ldots \theta^1 \epsilon_{t+m-1} + \epsilon_{t+m}, \tag{3.3.10}$$

for $m = 1, 2, \ldots$. From which the optimal m–period–ahead prediction is seen to be

$$E\left[y_{t+m} | y_t, y_{t-1}, \ldots,\right] = \theta^m y_t, \tag{3.3.11}$$

which is not a function of the whole past history. Hence, the process is Markov. Also not that the process is stable if $|\theta| < 1$. The form of equation (3.3.7) and (3.3.8) tells us that σ_t^2 is the *predicted* variance in the next time step, given the past sequences of predicted variances and the past realisations of the variances themselves. When $r = 0$ and $s = 2$, then

$$\sigma_t^2 = \sigma_0^2 + b_1 \epsilon_{t-1}^2 + b_2 \epsilon_{t-2}^2, \tag{3.3.12}$$

which becomes a Guassian process

$$\sigma_t^2 = \sigma_0^2 + b_1 \epsilon_{t-1}^2, \tag{3.3.13}$$

when $r = s = 0$.

Other versions of this model are

$$y_t = x_t^T \theta + \epsilon_t, \tag{3.3.14}$$

where x_t is the data model in RBFs and

$$y(t) = \mu + \sum_{i=1}^{n_y} \alpha_i y_{t-i} + \epsilon_t + \sum_{j=1}^{n_e} \beta_i \epsilon_{t-i}. \tag{3.3.15}$$

Also not that model (3.3.15) can be put in the form of model (3.3.14).

3.4 Estimating the correlation structure

The fitting procedure involves two stages:

1. Deciding the structure of the system whether it is moving average, autoregressive or both. *Note that here, we are assuming that RBFs is moving average* of the form

$$y_t = x_t^T \theta + \epsilon_t,$$

where s is the order of the moving average $MA(s)$.

2. Given s, say for the RBF, estimate the parameters, $\sigma_0, b_1, b_2, \ldots, \alpha_s$.

In passing, one might note that x_t is a vector of the same dimension as θ and it represents the scores of all the firing neurons in the hidden layer of RBFs. For $s = 2$, then

$$\sigma_t^2 = \sigma_0^2 + b_1 \epsilon_{t-1}^2 + b_2 \epsilon_{t-2}^2. \tag{3.4.1}$$

Since the disturbance ϵ_t satisfied these conditions, it is most convenient for the case of 2–lag, $r = 2$ to condition the estimation of the parameters on the first 2 observations. Let Y_t denote the vector of the observations obtained through date t :

$$Y_t = [y_t, y_{t-1}, \ldots, y_1, y_0, y_{-1}, a_t, a_{t-1}, \ldots, a_1, a_0, a_{-1}]$$

We have assume already that $v_t \sim N(0,1)$ and is independent of a_t and Y_t. Hence, the random variable $(y_t/a_t, Y_{t-1})$ is Gaussian with mean $a_t^T \theta$ and variance σ_t :

$$f(y_t/a_t, Y_{t-1}) = 1/(\sqrt{2\pi}\sigma_t) exp\left(\frac{-(y_t - a_t^T\theta)^2}{2\sigma_t^2}\right), \tag{3.4.2}$$

where

$$\sigma_t^2 = \sigma_0^2 + b_1(y_{t-1} - a_{t-1}^T\theta)^2 + b_2(y_{t-2} - a_{t-2}^T\theta)^2. \tag{3.4.3}$$

If we collect all the unknown parameters in a vector Θ then we can formulate the sample log likelihood condition on the first 2 observations and n data points:

$$L(\Theta) = \sum_{t=1}^{n} \log f(y_t/a_t, Y_{t-1}) \tag{3.4.4}$$

$$= -(n/2)\log(2\pi) - (1/2)\sum_{t=1}^{n}\log(\sigma_t^2) - (1/2)\sum_{t=1}^{n}(y_t - a_t^T\theta)^2/\sigma_t^2 \tag{3.4.5}$$

The log–likelihood function can now be maximized using optimisation techniques. This result of such an optimisation is what we now examine.

3.5 Correlation functions

In the discussion so far, we have assumed that the reader has some basic knowledge of the concept of stationarity and what we mean by autocorrelation function. The key role of stationarity here is that if we want to make predictions, clearly we must assume that something does not vary with time . In time series analysis of the sort that motivated this study, the goal is to predict series that typically is not deterministic but contain a random component. If this random component is stationary, in the sense of our above discussion, then we develop power techniques to forecast its future values.

In any case we define the autocovariance function (ACVF) of a stationary time series $\{y_t\}$ as

$$\gamma(h) = Cov(y_{t+h}, y_t), \quad h = \pm 1, \pm 2, \ldots,$$

and the autocorrelation function (ACF) of the series is defined similarly as the function

$$\rho(h) = \frac{\gamma(h)}{\gamma(0)}.$$

When we consider the prediction of future values of the series in terms of the past and present values the ACVF and ACF provide a useful measure of the degree of dependence between values of the time series at different times. Using the devices provided in (Davis and Vinter 1984) regarding what is understood as the "best " predictor of y_{t+h} given y_t we form the random variable $y_{t+h/t}$ known as the prediction of y_{t+h} given y_t. Assuming as they did that the process $\{y_t\}$ is a

stationary Guassian series, $y_t \sim N(\mu, \sigma^2)$, then

$$y_{t+h/t} \sim N(\mu + \rho(h)(y_t - \mu), \sigma^2(1 - \rho(h)^2))$$

From this result it follows that the function m of y_t that minimises $E\left[y_{t+h}, m(y_t)\right]^2$ is the conditional mean

$$m(y_t) = \mu + \rho(h)(y_t - \mu),$$

with the corresponding mean squared error

$$E\left[y_{t+h} - m(y_t)\right] = \sigma^2(1 - \rho(h)^2).$$

This computation shows that at least for stationary Gaussian processes, the prediction of y_{t+h} in terms of y_t is more accurate as $|\rho(h)|$ becomes closer to unity, making the corresponding mean square error to be closer to zero. Similar conclusions can be drawn as in (Davis and Vinter 1984), where they consider a more general problem of predicting as a function not only of y_t, but also of $y_{t-1}, y_{t-2}, \ldots$.

3.5.1 Partial correlation from Covariance matrix

The covariance matrix associated with the ACF at times $t_1, t_2, \ldots, t_n$, where n is the maximum lag, is

$$C_n = \sigma^2 \begin{bmatrix} 1 & \rho(1) & \rho(2) & \ldots & \rho(n-1) \\ \rho(1) & 1 & \rho(1) & \ldots & \rho(n-2) \\ \rho(2) & \rho(1) & 1 & \ldots & \rho(n-3) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \rho(n-1) & \rho(n-2) & \rho(n-3) & \ldots & 1 \end{bmatrix}. \qquad (3.5.1)$$

The matrix of the form (3.5.1), which have the property that the elements on the symmetric diagonals are identical, is called *Toeplitz* matrix. For a stationary stochastic process, it follows that these matrices C_n for $n = 2, 3, \ldots$ are positive semi–definite, that is, all principal minor of the determinant of C_n are positive or zero.

This result implies that the ACF of stationary time series satisfy all a wide spectrum of conditions. Let us look at one of these conditions, when the matrix C_n

singular. It may be verified that the determinant of C_k can be written as

$$\pi(k-1) = \frac{||C_k||}{|C_{k-1}|},$$

where $||C_k||$ is the determinant of the cofactor of the element in the first column and k^{th} row of C_k and $|.|$ specifies the determinant. The plot of $\pi(k)$ against k is called the *partial correlation function* (PCF) and has the property that if the process is autoregressive of order m, then

$$\begin{aligned} \pi(k) &\neq 0 \quad k \leq m, \\ \pi(k) &= 0 \quad k > m. \end{aligned}$$

Hence, the PCF provides a good model reduction tool to check whether an empirical time series can be fitted with an autoregressive process of a given order.

To see this, consider the case for $n = 3$. Then

$$\sigma^2 \begin{vmatrix} 1 & \rho(1) & \rho(2) \\ \rho(1) & 1 & \rho(1) \\ \rho(2) & \rho(1) & 1 \end{vmatrix} \geq 0. \tag{3.5.2}$$

Now assume that process is first order for which $\rho(2) = \rho(1)^2, \rho(3) = \rho(1)^3$, and so on so that

$$\pi(2) = \frac{\rho(2) - \rho(1)^2}{1 - \rho(1)^2}$$

is zero.

3.5.2 Testing for heteroscedastic using ACF and PCF

Our objective so far has been a singular one in which we seek for a series with not apparent deviation from stationarity, and in particular with no apparent trend or seasonality. This is the reason behind making the logarithmic transformation discussed earlier and the trend and seasonality remover algorithms discussed elsewhere in this thesis. Once it is done, the next important step is to model the residuals, (i.e. the estimated noise sequence) and investigate if there is any dependence between them. If there is no dependence between the residual sequences then the story ends with the estimated values of the mean and variance. Otherwise, we need to seek a more complex model that will account for the noise dependence. For large n, it can be shown that the ACF of an i.i.d. sequence of length n with finite variance are approximately i.i.d with distribution $N(0, 1/n)$.

From this proposition, we can draw a simple test of dependence by plotting, for up to lag 40, the ACF, checking for whether two, three or four outside the bounds $\pm 1.96/\sqrt{n}$, which is the 95% confidence bounds.

Although the relationship between the ACF and chi–square is too old, it is important to realise that there is a very important relationship between a quadratic form $Q = x^2$, with $x \sim N(0, \sigma)$ and the chi–squares distributions, see appendix and (Zacks 1980). This result motivates the test statistics

$$Q = n \sum_{i=1}^{h} \hat{\rho}^2(i), \tag{3.5.3}$$

where $\hat{\rho}(i)$ is the estimated ACF at lag i. It turns out that Q chi–squares distribution. A better estimate is given by (Ljung and Box) (see Box *et al* 1994)

$$Q = n(n+2) \sum_{i=1}^{h} \hat{\rho}^2(i)/(n-i) \tag{3.5.4}$$

which is a better approximation to the chi–square assumption than in (3.5.3). Another test is given by

$$Q = n(n+2) \sum_{i=1}^{h} \hat{\rho}_{yy}^2(i)/(n-i), \tag{3.5.5}$$

where the sample ACF is replaced with the sample ACF of the squared data (Box *et al* 1994). This test is known as McLeod–Li test, after the authors who designed the test.

Look at the electricity consumption and note that there is no long run mean level μ upon which the time series evolve. This is because of *transient effect, i.e.* a kind of nonstationary series. To apply the method of our framework we need a stationary series. Consider the following nonlinear transformation

$$y_t \doteq \log \frac{y(t+1)}{y(t)} = \log y(t+1) - \log y(t), \tag{3.5.6}$$

which acts on the time series $y(t)$, producing the nonlinear time series $y_t, y_{t-1}, \ldots,$. In contrast, y_t appears to be quite stable over time, and the transformation from time series to logarithmic sequences has produced a stationary time series, as shown in figure (3.5). This motivates us to write (3.3.15)

$$y_t = \mu + \epsilon_t \tag{3.5.7}$$

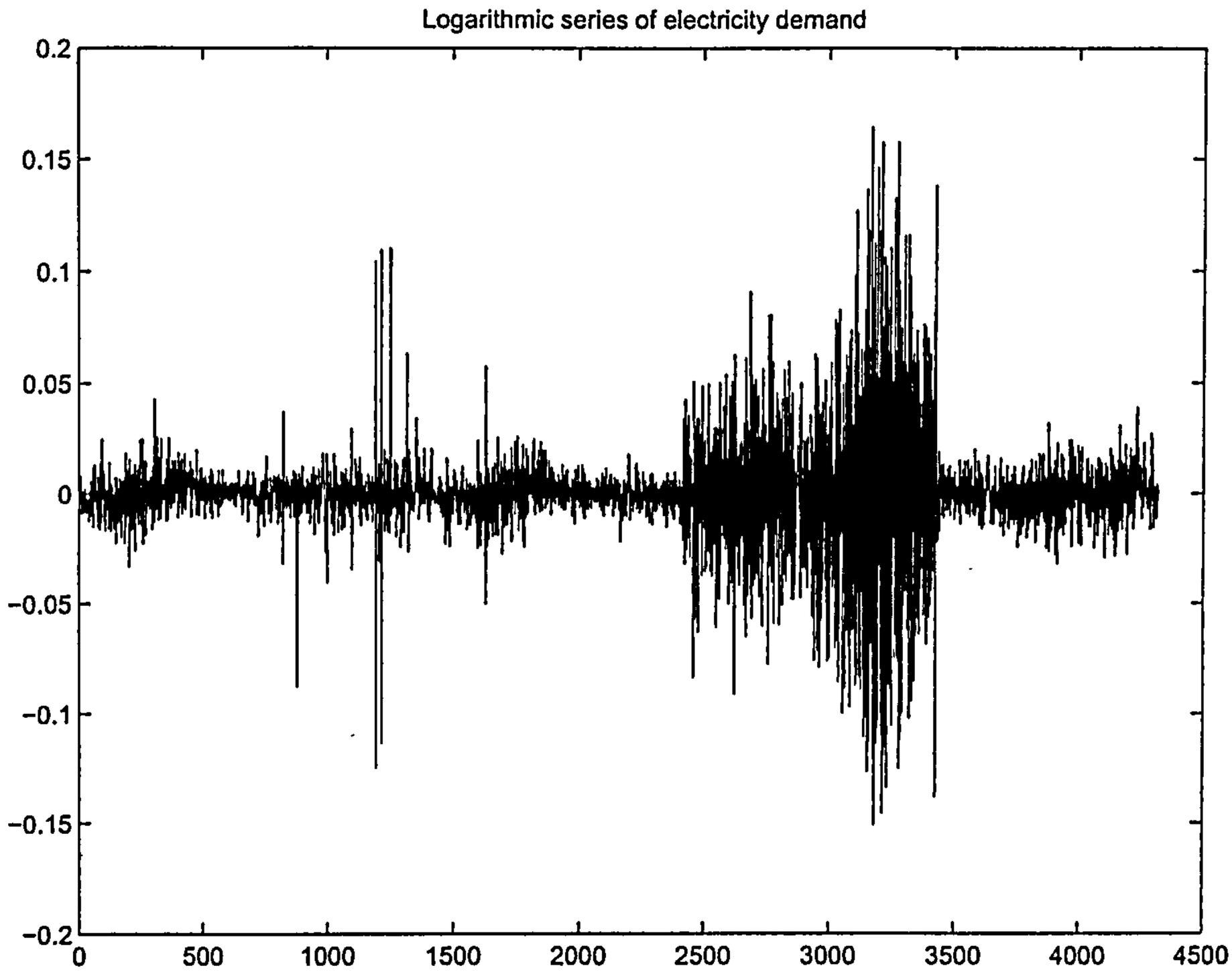


Figure 3.5: Electricity demand as Logarithmic series

where $n_y = 0$ and $n_e = 1$. The relation (3.3.8) is also given as

$$\sigma_t^2 = \sigma_0^2 + a_1\sigma_{t-1}^2 + +b_1\epsilon_{t-1}^2. \tag{3.5.8}$$

Equation (3.5.7) is the conditional mean model of the process and (3.5.8) is the conditional variance model.

Thus, analogous to Doob decomposition, the random variable y_t is made of two components, the predictable component μ and the uncorrelated white noise disturbance component ϵ_t, the martingale part. Similarly, equation (3.5.8) is the conditional variance model, leading to a variance prediction σ_t^2, composing of a constant predictable σ_0, a last time–step prediction σ_{t-1}^2 and a martingale part ϵ_{t-1}^2.

Note: What this entails that by looking at the graph of the process $\{y_t\}$ and its autocorrelation function, we may conclude that the process is a realisation of an IID noise, i.e. the process is white noise. However, the series may not be a realisation of an IID noise because we have only considered the process up and including the second order.

Now, suppose that we consider the second order process $\{y_t^2\}$, looking at the graph

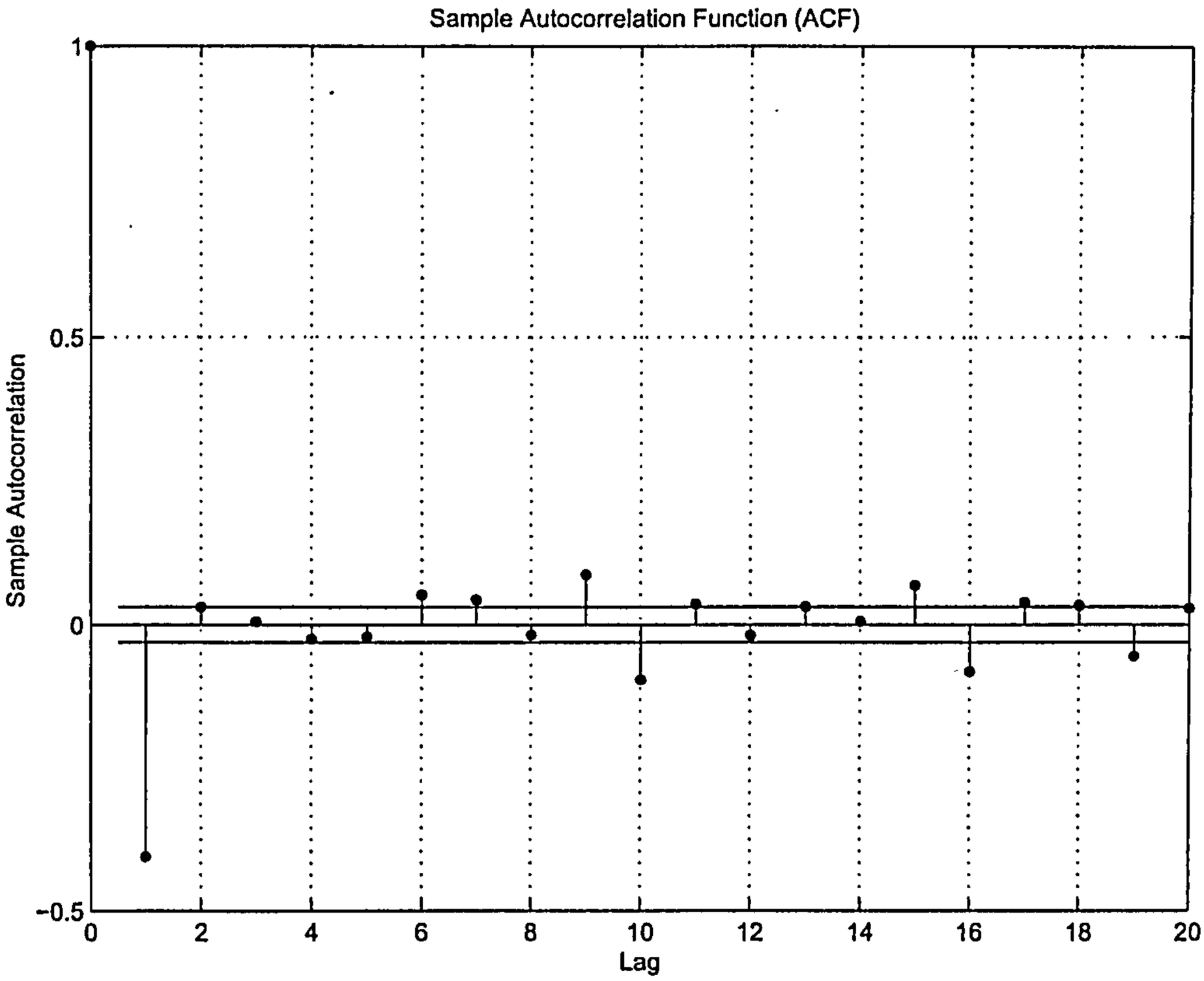


Figure 3.6: ACF for the time series

of its realisation and the sample autocorrelation function will let us know whether the process is IID or not. Returning to our logarithmic series y_t for electricity demand, you can check qualitatively for correlation in the raw series by calling the functions computing the autocorrelation ACF and partial-autocorrelation (PACF) function, respectively. Figure (3.6) displays the sample ACF of the series, along with the upper and lower standard deviation confidence bounds, based on the assumption that all autocorrelations are zero beyond lag zero.

In addition, figure (3.7) displays the sample PACF with upper and lower confidence bounds.

From these two figures, there is some real indication that we need to use some correlation structure in the conditional mean. This observation is not very conclusive, as the graphs strongly indicate a white noise process. This follows from the assumption that is inherent in white noise –that their ACF should die down to zero after zero lag. Also, notice the similarity between the graphs.

However, as preliminary identification tools, the ACF and PACF provide some indication of the broad correlation characteristics of the series. From figure (3.6)

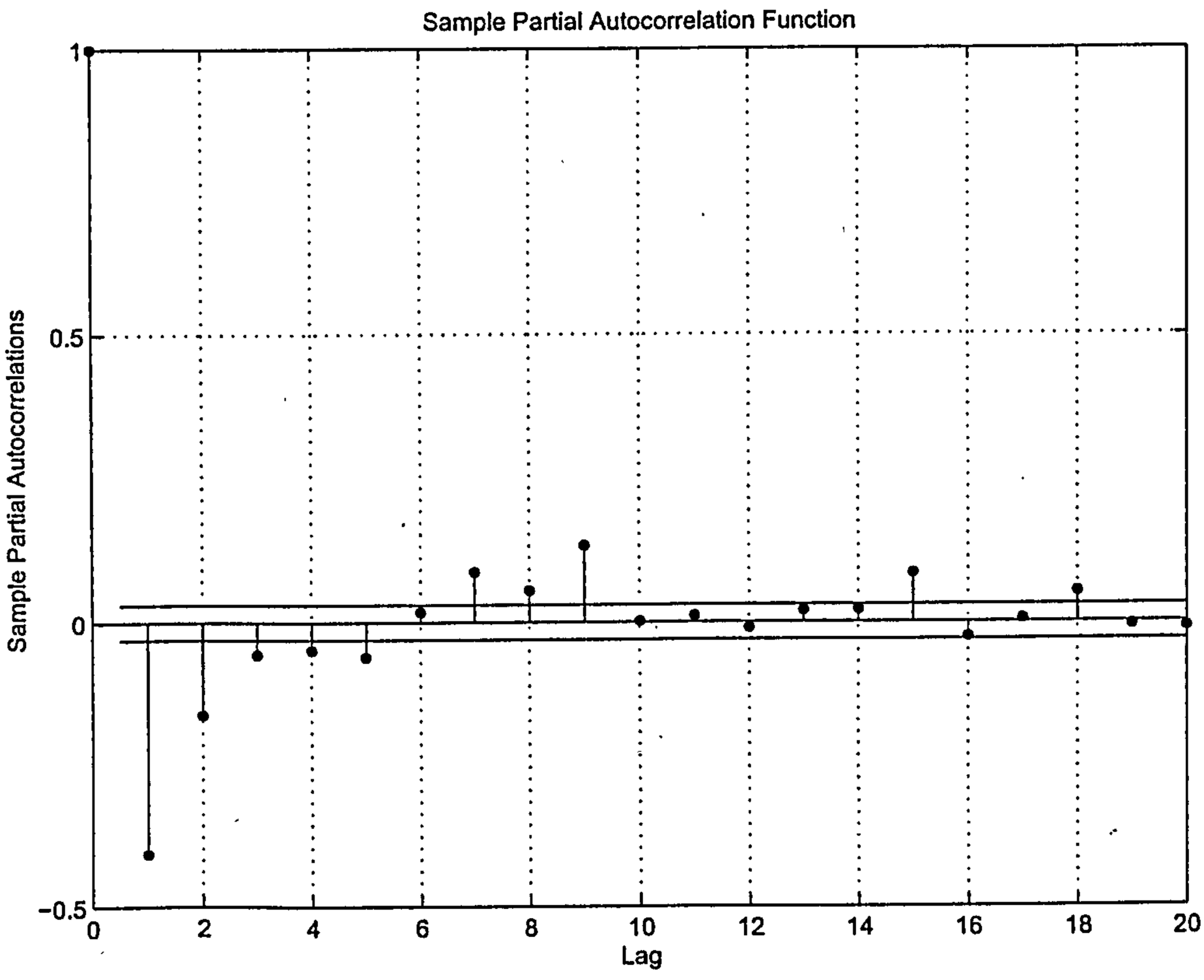


Figure 3.7: PCF for the time series

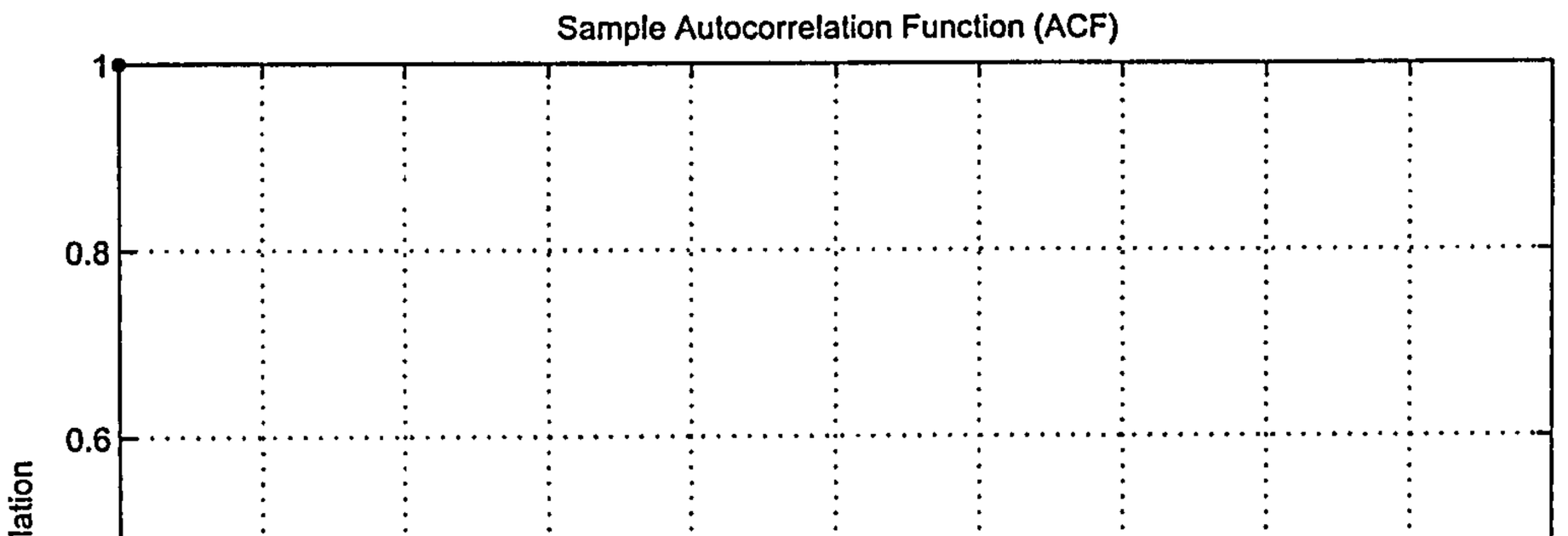


Figure 3.8: ACF for the squared– time series

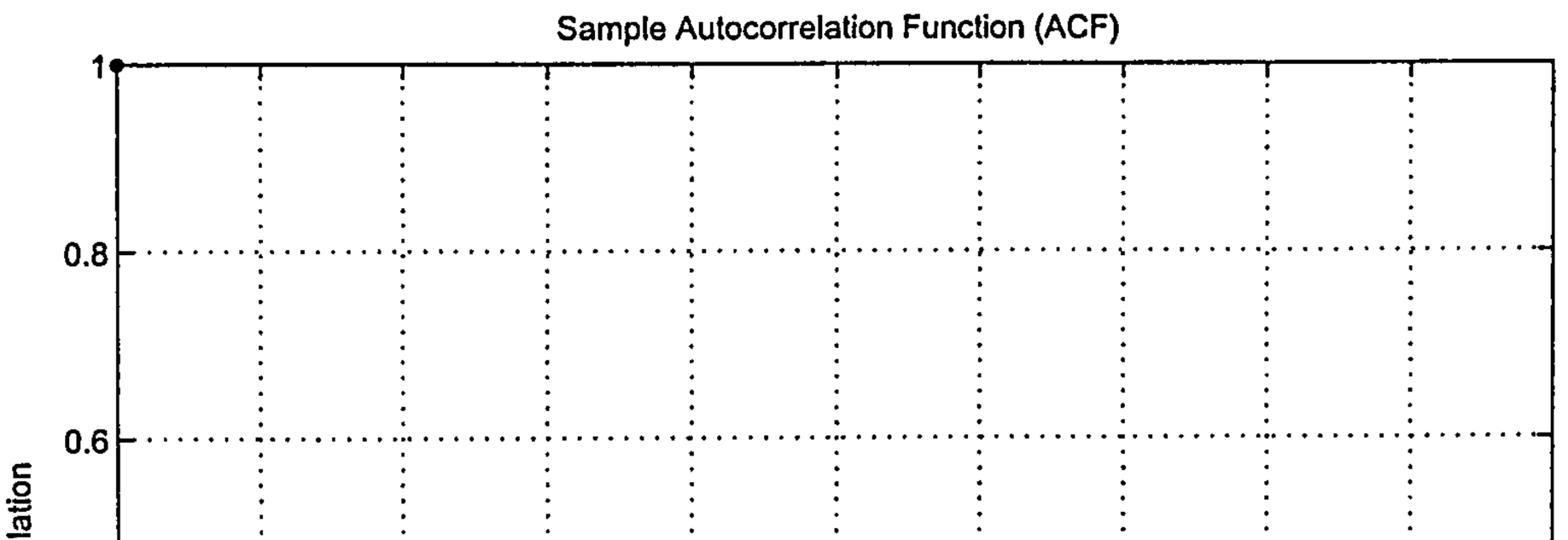


Figure 3.9: PCF for the squared– time series

and (3.7) , we ignore any correlation structure in the conditional mean.

Although the ACF of the observed series $\{y_t\}$ exhibits little correlation, the ACF of the squared returns may still indicate significant correlation and persistence in the second-order moments. we check this by plotting the ACF of the squared process$\{y_t^2\}$. Figure (3.8) shows that, although the series $\{y_t\}$ themselves are largely uncorrelated, the variance process exhibits some correlation. This is consistent with what is known about the theory of these processes (Engle 1982). Note that the ACF shown in figure (3.8) appears to die out slowly, revealing the possibility of a variance process close to being nonstationary.

3.6 Expiements on volatility

The presence of volatility, shown in the previous analysis, indicates that correlated modelling is appropriate. Assume the model described in (3.5.7) and (3.5.8) and use the logarathmic transformation of the electricity data. Thus, the number of variables to be estimated is 4, forming a parameter vector $\Theta = (\mu, \sigma_0, a_1, b_1)$.

Although simplistic, has several benefits against RBFs model below:

1. It represents a parsimonious model that requires one to estimate only four parameters $(\mu, \sigma_0, a_1, b_1)$. According to a saying in machine learning, the fewer parameters to estimate, the less that can go wrong. Elaborate models often fail to offer real benefits when forecasting.

2. The simple nonlinear correlation model captures most of the variability in most series.

3. Small lags for r and s are common in demand applications. Typically, lags of $(1,1),(2,1),(1,2)$ models are adequate for modelling volatilities even over long sample periods.

We will like to print the diagnostic, optimization, and summary information below. We did the optimisation using Matlab (See Mathworks). We set up the following optimisation:

1. Number of variables: $\Theta = (\mu, \sigma_0, a_1, b_1)$.[1]

[1]Our notation may cause confusion, $\sigma_0 \neq \sigma_t, t = 0$.

2. Functions Objective: Log–likelihood function of $\Theta = (\mu, \sigma_0, a_1, b_1)$.

3. Gradient: finite-differencing

4. Hessian: (or Quasi-Newton)

5. Nonlinear constraints:

$$a_1 + b_1 < 1,$$

$$\sigma_0 > 0,$$

$$a_1 > 0,$$

$$b_1 > 0.$$

The first constraint(a stationarity constraint) is necessary and sufficient for the existence of a finite, time–independent variance of the innovations process ϵ_t. The remaining constraints are sufficient to ensure that the conditional variance σ_t is strictly positive

6. Gradient of nonlinear constraints: finite-differencing

Constraints

7. Number of nonlinear inequality constraints: 0

8. Number of nonlinear equality constraints: 0

9. Number of linear inequality constraints: 1

10. Number of linear equality constraints: 0

11. Number of lower bound constraints: 4

12. Number of upper bound constraints: 0.

Iter	F-count	$f(x)$	constraint	Step-size
1	5	-11457.4	$-4.983e-005$	1
2	32	-11457.4	$-4.983e-005$	$4.77e-007$
3	39	-12398.1	$-2.492e-005$	0.5
4	50	$-12401.2-3.161e-005$	0.0313	-601
5	57	$-12598.7-1.581e-005$	0.5	-197
6	64	$-12685.2-7.903e-006$	0.5	-218
7	73	$-12695.3-6.915e-006$	0.125	-141
8	80	$-12756.7-3.457e-006$	0.5	-840
9	91	$-12768-3.454e-006$	0.0313	-81.4
10	98	$-12802.8-1.727e-006$	0.5	-150
11	108	$-12805-2.061e-006$	0.0625	-27.5
12	115	$-12810.4-1.031e-006$	0.5	-17.4
13	121	-12813.10	1	-2.94
14	127	-12814.50	1	-16.9
15	138	$-12814.8-1.108e-006$	0.0313	-2.09
16	158	$-12814.8-1.107e-006$	$-6.1e-005$	-0.065
17	164	$-12814.8-1.065e-006$	1	-0.0276
18	171	$-12814.8-1.078e-006$	0.5	-0.00853
19	177	$-12814.8-1.07e-006$	1	-0.00236
20	183	$-12814.8-1.074e-006$	1	-0.00103
21	189	$-12814.8-1.072e-006$	1	-0.000425
22	196	$-12814.8-1.072e-006$	0.5	-0.000124
23	202	$-12814.8-1.072e-006$	1	-0.00164
24	222	$-12814.8-1.072e-006$	$-6.1e-005$	-0.000857

Optimisation terminated successfully. Figure (3.11) shows the plot of the innovation (what we called the martigale error sequences), the standard deviations and the return (what we called the logarithmic transformed data).

The optimisation gives the following result
$\mu = 1.9983e-004, \sigma = 1.0718e-006, a_1 = 0.9303, b_2 = 0.0680.$

3.6.1 Experiment on finite horizon prediction

In this experiment we compute minimum–mean–square–error (MMSE) forecasts of the conditional mean and conditional standard deviation of the logarithmic series y_t in each period over a user–specified forecast horizon.

Now, we want to compute the conditional mean and standard deviation forecasts for the electricity demand using the model equation (3.5.7) and the estimated parameter estimates.

The conditional standard deviation σ_1 is 0.0081 and the conditional mean is 0 $1.9983e-004$. For asymptotic behaviour, we need to forecast for more than a single period. Here we forecast the conditional standard deviation and mean in each period of a 10-period forecast horizon.

σ_t	μ
0.0081	0.0002
0.0082	0.0002
0.0082	0.0002
0.0083	0.0002
0.0083	0.0002
0.0084	0.0002
0.0085	0.0002
0.0085	0.0002
0.0086	0.0002
0.0086	0.0002.

The results show that the model (3.5.7) forecastof the conditional mean is always = 0.0002. This is true for any prediction horizon because the expected value of any innovation, ϵ_t, is 0.

In contrast, the conditional standard deviation forecast changes from period to period and approaches the unconditional standard deviation of ϵ_t, given by the square root of

$$\sigma^2 = E\left[\epsilon_t^2\right] = \frac{\sigma_0}{1 - \sum a_i - \sum b_i}. \tag{3.6.1}$$

From this example, we can now calculate the unconditional standard deviation of ϵ_t as 0.0252. We plot the conditional standard deviations derived from the fitted series. The plot reveals that the most recent values of σ_t fall below this long-run asymptotic. The fact that the most recent values of σ_t fall below 0.0252. This means that the long–range prediction of σ_t approaches this value from below. We confirm this by forecasting the standard deviations over 1000 periods, then plotting the forecasts (dashed) and asymptotic value (solid) on the same figure (3.12).

The graph shows that it takes a very long time for the prediction to reach its steady-state value. This is consistent with the high degree of persistence in the

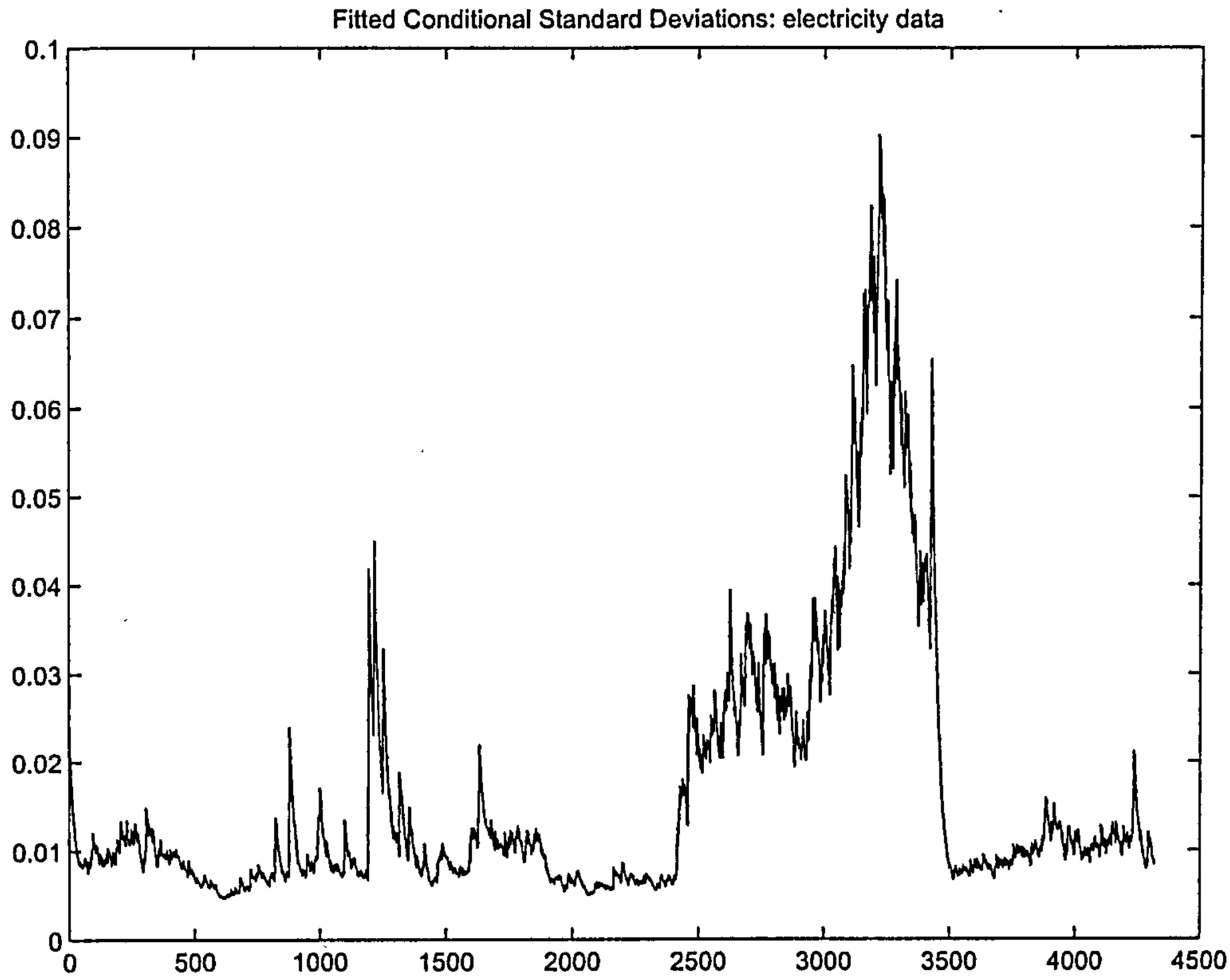


Figure 3.10: Forecasting standard deviation

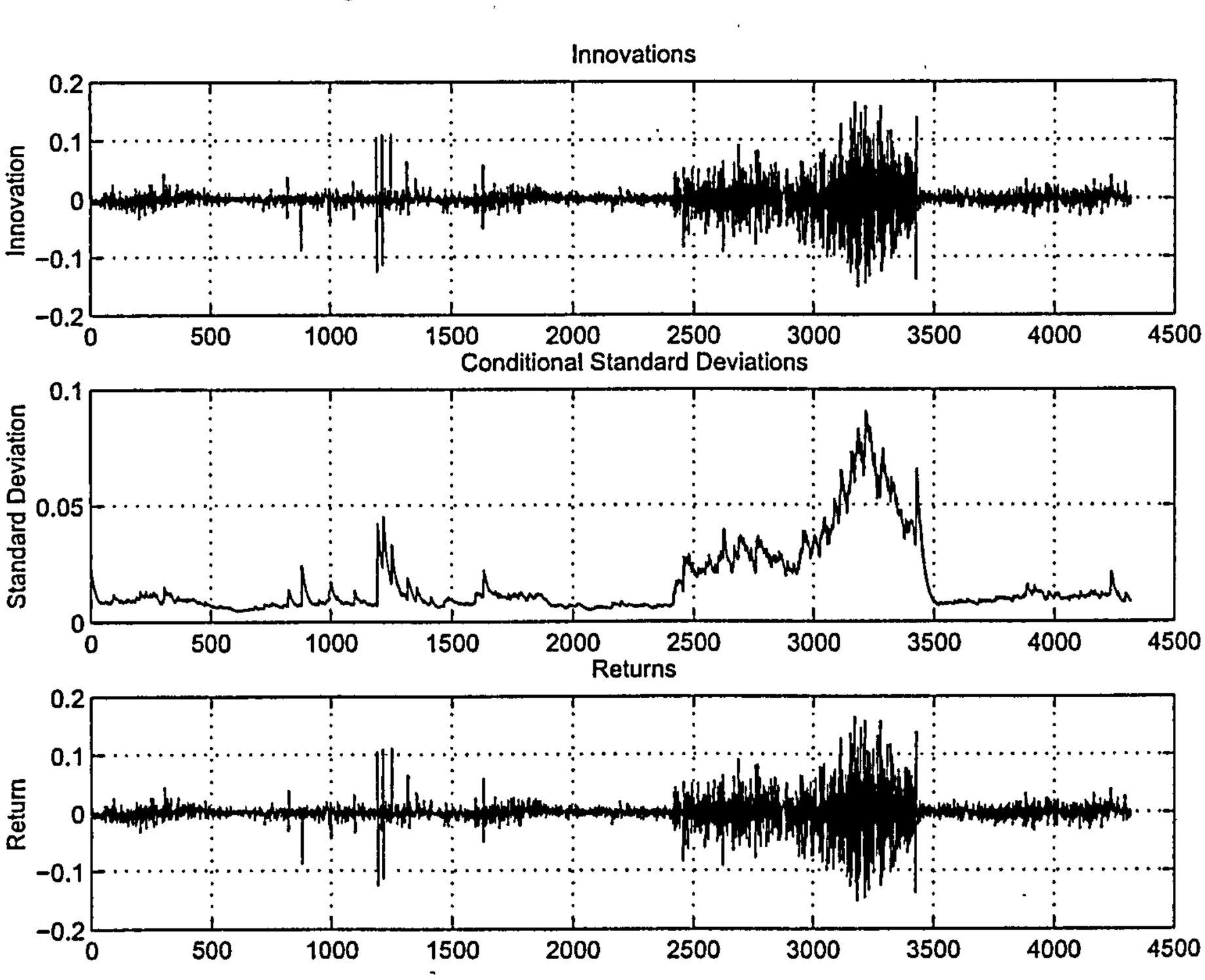


Figure 3.11: Plots

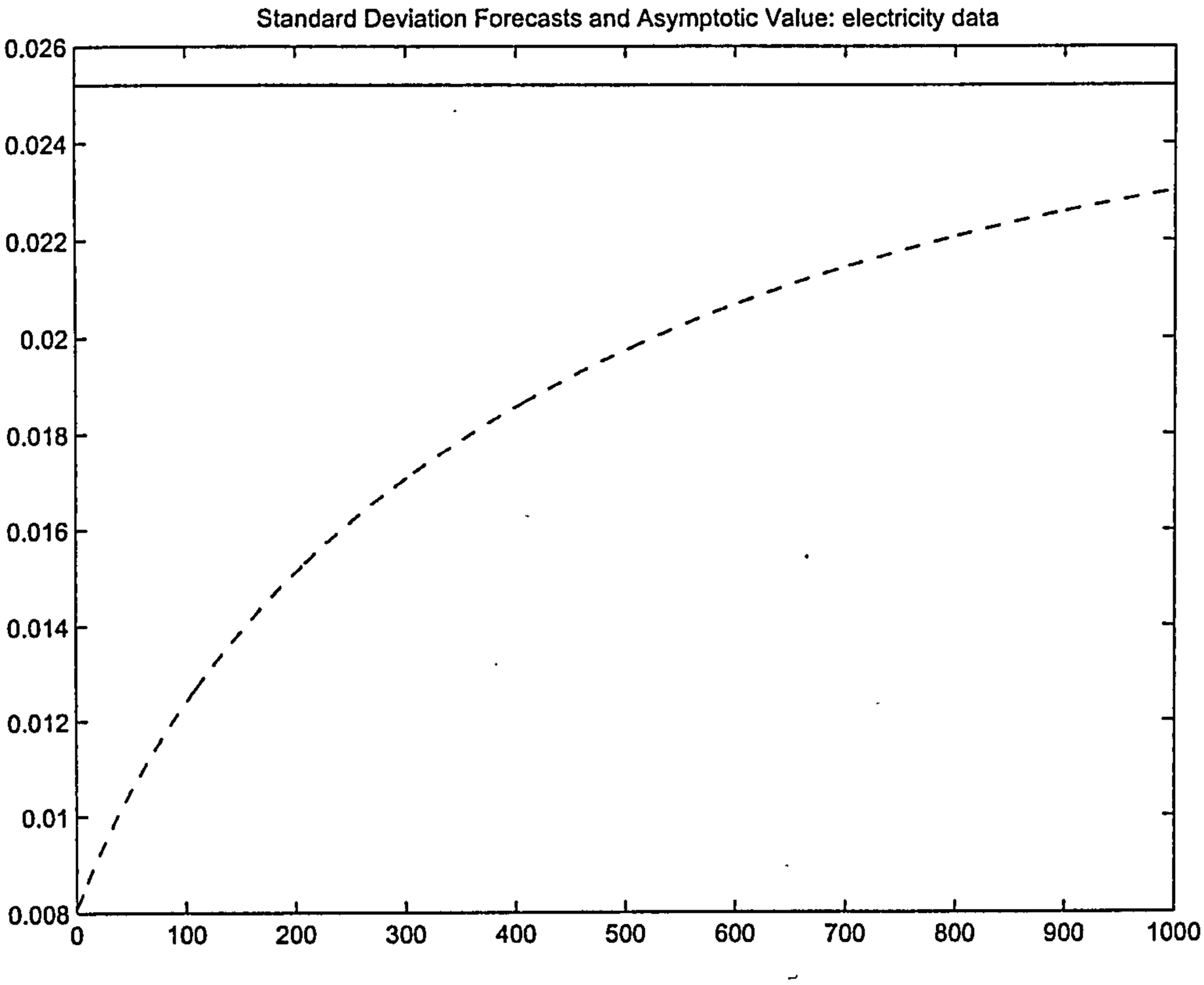


Figure 3.12: Forecasting standard deviation in the long range volatility process for the electricity data.

3.7 Experiments: Correlated errors

Now, consider the system

$$y_{t+1} = \mu + \alpha_1 y_{t-1} + \alpha_2 y_{t-2} + \epsilon_t + \beta_1 \epsilon_{t-1}. \tag{3.7.1}$$

and the error variance

$$\sigma_t^2 = \sigma_0^2 + a_1 \sigma_{t-1}^2 + b_1 \epsilon_{t-1}^2,$$

using again the electricity data. Thus, we have:

1. Number of variables $\Theta = (\mu, \alpha_1, \alpha_2, \beta_1, \sigma_0, a_1, b_1)$ is 7.
2. Objective function is the log–likelihood function of the 7 parameters.
3. For the gradient we use a finite-differencing of the log–likelihood.
4. For Hessian we use Quasi-Newton optimisation.

5. Gradient of nonlinear constraints we use finite–differencing.

6. Number of nonlinear inequality constraints: 1

7. Number of nonlinear equality constraints: 0

8. Number of linear inequality constraints: 1

9. Number of linear equality constraints: 0

10. Number of lower bound constraints: 7

11. Number of upper bound constraints: 0

We specify all these in the Matlab optimisation routine and obtain the following results:

Parameter	Value	Standard Error	Statistic
μ	17.057	3.9426	4.3262
α_1	0.75409	0.13691	5.5079
α_2	−0.086167	0.15006	−0.5742
β_1	−0.94784	0.041416	−22.8857
σ_0	47.279	41.213	1.1472
a_1	0.070195	0.37768	0.1859
b_1	0.62023	0.38713	1.6021

Now let us consider the model

$$y_{t+1} = \mu + \alpha_1 y_{t-1} + \alpha_2 y_{t-2} + \epsilon_t. \tag{3.7.2}$$

and the error variance

$$\sigma_t^2 = \sigma_0^2 + a_1 \sigma_{t-1}^2 + b_1 \epsilon_{t-1}^2$$

with no β terms as in the regression case. We need to see if the results are the same.

Parameter	Value	Error	T Statistic
μ	55.652	9.2391	56.0236
α_1	−0.0406	0.1429	−0.2842
α_2	−0.04425	0.12805	−0.3324
σ_0	14.237	693.59	0.0205
a_1	0.89692	5.0138	0.1789
b_1	0	0.0692	0.0000

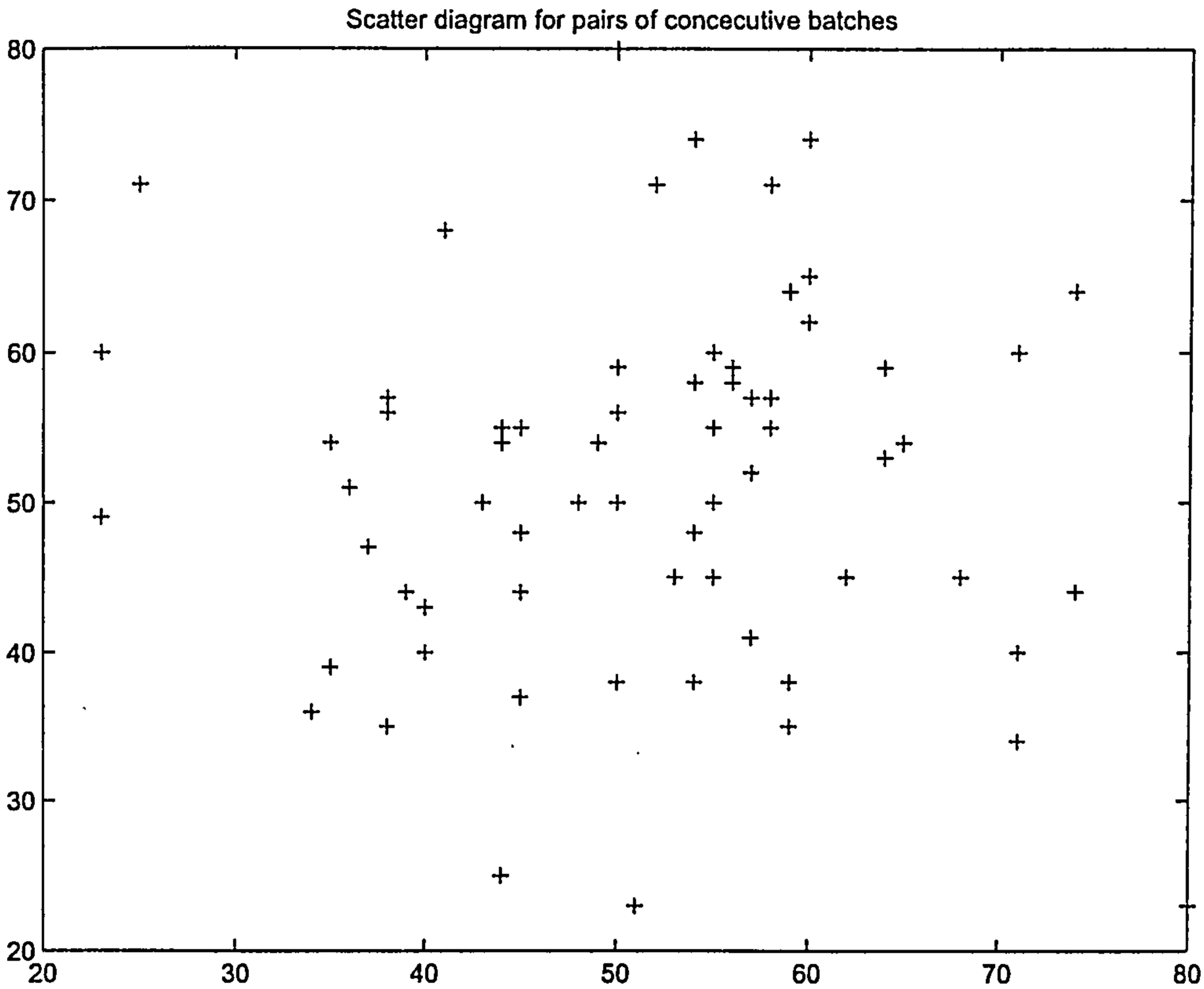


Figure 3.13: Batching data

3.8 Experiment on regression using batch data

To illustrate the use of a regression matrix, fit the production batch data, to an $AR(2)$ process in the mean, to a regression model with two explanatory variables. The regression matrix consists of the first- and second-order lags of the simulated series y.

$$y_{t+1} = \alpha_1 y_{t-1} + \alpha_2 y_{t-2} + \epsilon_{t+1}. \tag{3.8.1}$$

and error variance

$$\sigma_t^0 = \sigma_0^2 + a_1 \sigma_{t-1}^2.$$

Figure (3.13) shows the scatter diagram for consecutive batches (y_{t+1}, y_t) for the data of the production plant. As shown the points fall mainly in the northwest and southeast corners of the diagram, bringing out the negative dependent between neighbouring batches evident in figure (3.14).

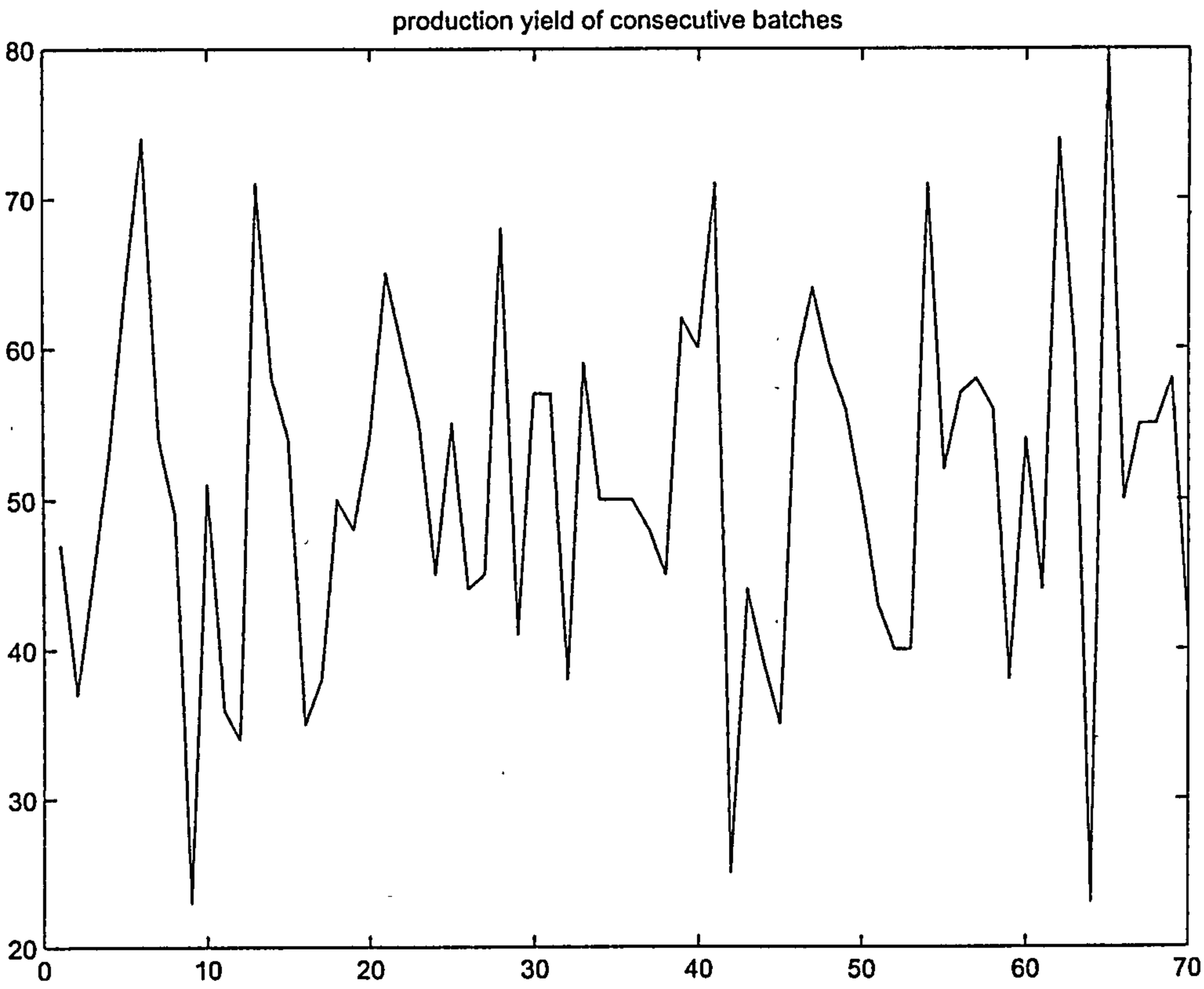


Figure 3.14: Batching data

Parameter	Value	Standard Error	T Statistic
μ	55.558	9.1405	6.0782
α_1	−0.037682	0.13451	−0.2801
α_2	−0.048243	0.12591	−0.3832
σ_0	9.5458	612.48	0.0156
a_1	0.93074	4.4283	0.2102

Repeat the experiment under small change of optimisation conditions, assuming now that the model is of the form

$$\sigma_t^0 = \sigma_0^2 + a_1\sigma_{t-1}^2 + b_1\epsilon_{t-1}^2$$

and obtain the similar results:

Parameter	Value	Error	T Statistic
μ	55.209	9.2899	5.9429
α_1	−0.034235	0.14465	−0.2367
α_2	−0.045077	0.12807	−0.3520
σ_0	14.066	1027.7	0.0137
a_1	0.89828	7.4223	0.1210
b_1	0	0.068963	0.0000

3.9 Further Experiment with autoregressive and correlated errors

Now, consider the system

$$y_{t+1} = \mu + \alpha_1 y_{t-1} + \alpha_2 y_{t-2} + \epsilon_t + \beta_1 \epsilon_{t-1}. \tag{3.9.1}$$

and the error variance

$$\sigma_t^2 = \sigma_0^2 + a_1\sigma_{t-1}^2 + b_1\epsilon_{t-1}^2$$

Thus, we have:

1. Number of variables $\Theta = (\mu, \alpha_1, \alpha_2, \beta_1, \sigma_0, a_1, b_1)$ is 7.
2. Objective function is the log–likelihood function of the 7 parameters.
3. For the gradient we use a finite-differencing of the log–likelihood.

4. For Hessian we use Quasi-Newton optimisation.

5. Gradient of nonlinear constraints we use finite–differencing.

6. Number of nonlinear inequality constraints: 1

7. Number of nonlinear equality constraints: 0

8. Number of linear inequality constraints: 1

9. Number of linear equality constraints: 0

10. Number of lower bound constraints: 7

11. Number of upper bound constraints: 0

We specify all these in the Matlab optimisation routine and obtain the following results:

Parameter	Value	Standard Error	Statistic
μ	17.057	3.9426	4.3262
α_1	0.75409	0.13691	5.5079
α_2	−0.086167	0.15006	−0.5742
β_1	−0.94784	0.041416	−22.8857
σ_0	47.279	41.213	1.1472
a_1	0.070195	0.37768	0.1859
b_1	0.62023	0.38713	1.6021

Now let us consider the model

$$y_{t+1} = \mu + \alpha_1 y_{t-1} + \alpha_2 y_{t-2} + \epsilon_t. \tag{3.9.2}$$

and the error variance

$$\sigma_t^2 = \sigma_0^2 + a_1 \sigma_{t-1}^2 + b_1 \epsilon_{t-1}^2$$

with no β terms as in the regression case. We need to see if the results are the same.

Parameter	Value	Error	T Statistic
μ	55.652	9.2391	56.0236
α_1	−0.0406	0.1429	−0.2842
α_2	−0.04425	0.12805	−0.3324
σ_0	14.237	693.59	0.0205
a_1	0.89692	5.0138	0.1789
b_1	0	0.0692	0.0000

3.9.1 Estimating mean from insurance daily claims

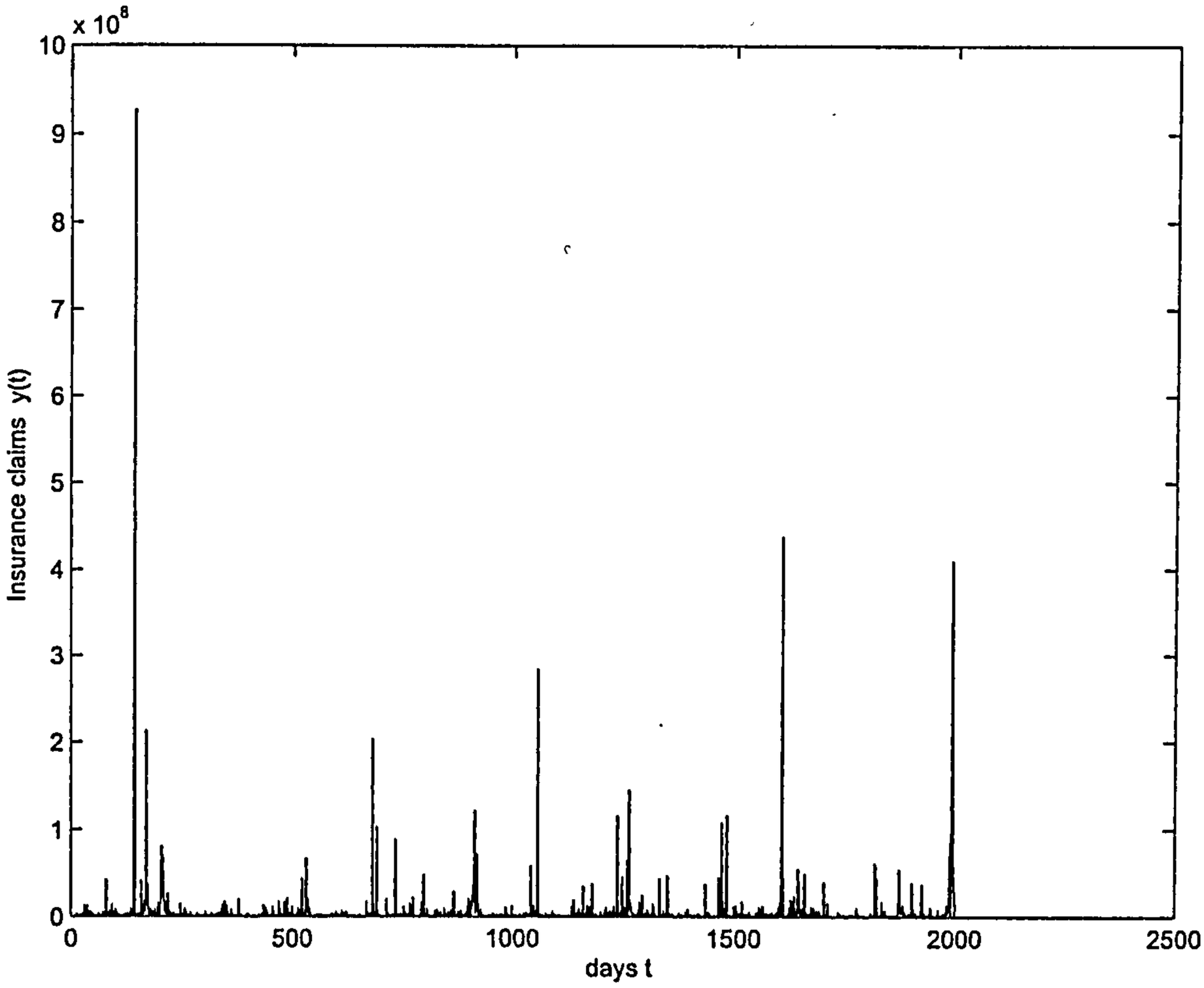


Figure 3.15: Daily insurance claims

With respect to the errors that is perturbing the RBFs follow an autoregressive scheme can be written as

$$y_{t+1} = \mu + \sum_{i=1}^{n} \alpha_i y_{t-i} + \epsilon_t \sum_{i=1}^{m} \beta_i \epsilon_{t-i}, \tag{3.9.3}$$

where the variance of the noise follows a similar recurrence scheme of the form

$$\sigma_t^2 = \sigma_0^2 + \sum_{i=1}^{p} \alpha_i \sigma_{t-i}^2 \sum_{i=1}^{q} b_i \epsilon_{t-i}^2, \tag{3.9.4}$$

for some integer values n, m, p, q that can be determined from model reduction of chapters (2). When p is non–zero then the above model is transient. The objective of this section is to find the parameters of the model such that we can obtain the long run values of y_∞ and σ_∞. It can be shown that if the positivity conditions specified about the coefficients of the above systems are met then the long term

solutions are given as

$$E[y_t] = \frac{\mu}{1 - \alpha_1 - \alpha_2 - \alpha_3 - \ldots - \alpha_n}. \tag{3.9.5}$$

The unconditional mean y_t is constant. From control theoritical point of view the above statement is true if and only if the L_1 process $\{y_t\}$ is covariance stationary[2] . Such stationarity occurs if the roots of the characteristic equations

$$1 - \sum_{i=1}^{n} \alpha_i z^i = \tag{3.9.6}$$

lies outside the unit circle.

But, we are forced to estimate the variance of y_t because as you see in the plots of the daily claims, which is claimed recorded and charged overnight for the company with transfer of money from one bank to the others. This probably explains why the daily claims vary from one day to another, making it volatile and changes in variance becomes very important. The insurance consultant is required to look at changes in the variance so as to make changes in the premium rating and pricing of insurance sales and other assets. When these conditions are met, the unconditional variance

$$Var[y_t] = \frac{\sigma_0^2}{1 - a_1 - a_2 - a_3 - \ldots - \alpha_p - b_1 - b_2 - b_3 - \ldots - b_p}. \tag{3.9.7}$$

What remains to be done is just the maximum likelihood estimate of the parameters and from these we can estimate the steady state mean.

The results, shown in Figure (3.16), are the log of the daily insurance claims. Notice the presence of volatility clustering in the series, the variance is changing. Now, assume that the claims data follow an autoregressive system

$$y_{t+1} = \mu + \alpha_1 y_t + \alpha_2 y_{t-1} + \epsilon_t + \beta_1 \epsilon_{t-1}, \tag{3.9.8}$$

$$\sigma_t^2 = \sigma_0^2 + a_1 \sigma_{t-1}^2 + b_t \epsilon_{t-1}^2. \tag{3.9.9}$$

Using similar optimisation, we have the results

$$\mu = -0.0025, \alpha_1 = 0.2648, \alpha_2 = 0.0364, \beta_1 = -0.9215, \sigma_0 = 1.7247, a_1 = 8.4455e{-}008, b_1 = 0.1275$$

[2]An assumption that is taken for granted by some researchers who simulate quueieng systems.

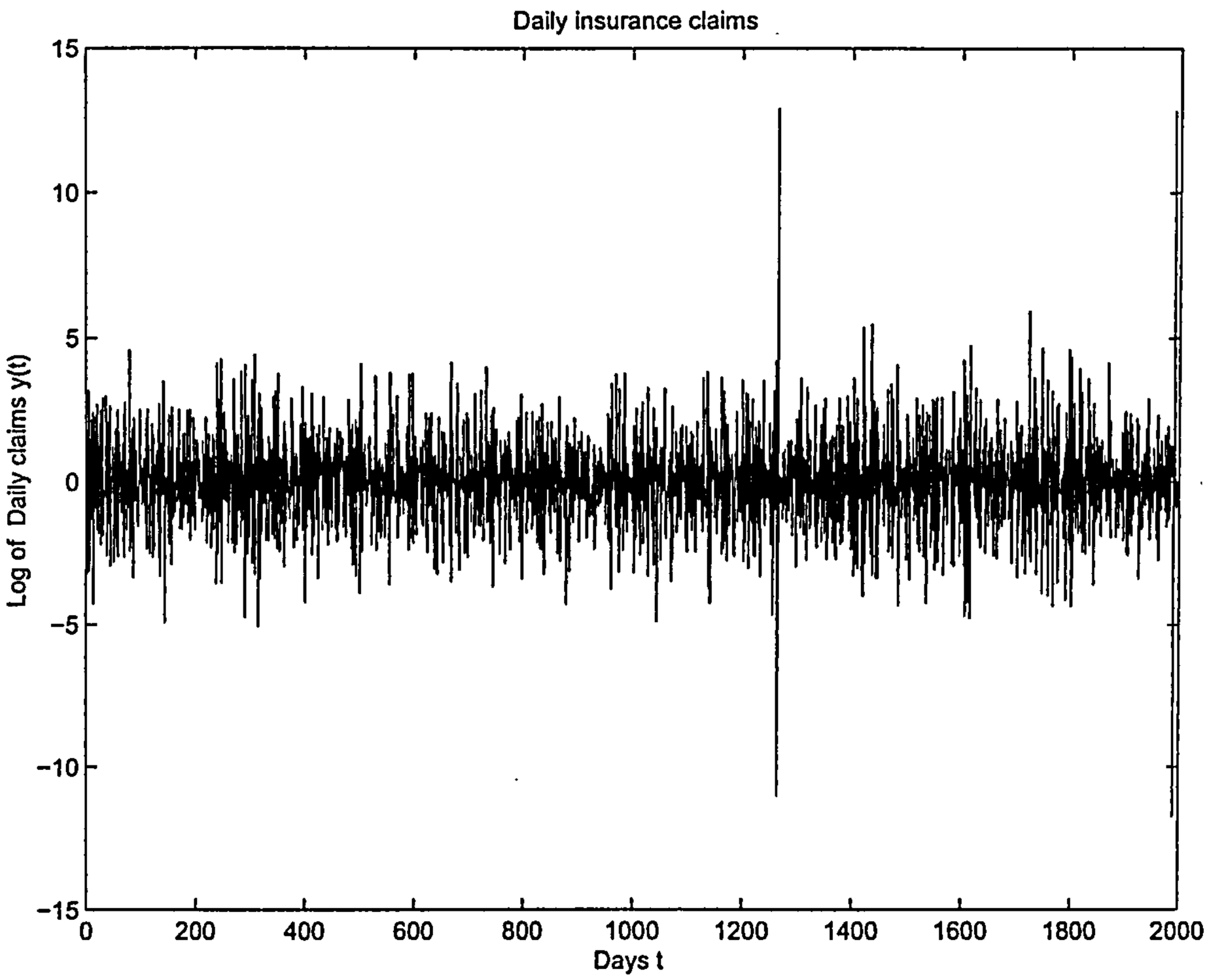


Figure 3.16: Log of daily insurance claims

From this we can deduce that stationary mean value y_∞.

3.10 Experiment on model reduction

We have consider ways of finding the maximum likelihood estimate of parameters as an optimisation problem on the likelihood function $L(\Theta)$. We did this by relying on an assumption that the data are strictly stationary and that neither the estimate and the true parameter lies at the boundary. We use the batch data set and consider the system

$$y_{t+1} = \mu + \epsilon_t + \beta_1 \epsilon_{t-1}. \tag{3.10.1}$$

and the error variance

$$\sigma_t^2 = \sigma_0^2 + a_1 \sigma_{t-1}^2 + b_1 \epsilon_{t-1}^2 \tag{3.10.2}$$

$$\sigma_t^2 = \sigma_0^2 + a_1 \sigma_{t-1}^2 + a_2 \sigma_{t-2}^2 + b_1 \epsilon_{t-1}^2. \tag{3.10.3}$$

These two systems definitely have two log–likelihood functions, one denoted as $L(\Theta_1$ where the term $a_2 = 0$ and the other, denoted as $L(\Theta_2$, where there is no

restriction on a_2. Since the test of hypothesis here involves only one restriction, there is just one degree of freedom in the difference $L(\Theta) = (L(\Theta_2) - L(\Theta_1)$. It turns out from the quadratic nature of the test that

$$L(\Theta) \sim \chi(1),$$

has a chi–squares distribution.

In this context, the unrestricted model(3.10.2 serves as the alternative hypothesis (i.e., the hypothesis the example gathers evidence to support), while the restricted model (3.10.3 serves as the null hypothesis (i.e., the hypothesis the example assumes is true, lacking any evidence to support the alternative).

Thus, what is called in statistics the likelihood ratio test (Brockwell and Davis 1996) is asymptotically Chi-square distributed with degrees-of-freedom equal to the number of restrictions imposed. The proof of this assertion is easy and can be found from the relationship between quadratic forms of Gaussian random variables and their distributions.

In this experiment the restricted model has the following results:

rameter	Value	Standard Error	T Statistic
μ	51.145	1.4265	35.8536
σ_0	14.111	1353.1	0.0104
a_1	0.89869	9.7011	0.0926
b_1	0	0.059749	0.0000.

Model (3.10.3) produces the following results:

Parameter	Value	Standard Error	Statistic
μ	51.066	1.4756	34.6062
σ_0	68.818	135.42	0.5082
a_1	0	0.58056	0.0000
a_2	0.31164	0.64932	0.4799
b_1	0.20407	0.21694	0.9406.

Of the two models associated with the same series: The model (3.10.2) is a restricted model. That is, you can interpret it as model (3.10.3) with the restriction that $a_2 = 0$. The more elaborate model (3.10.3) is an unrestricted model. We test this statistic at significance level 0.05 obtain the result that the first model is a better model .

3.10.1 Information criteria

Akaike information criterion (AIC) is a method explained in (Dàvis an d Vinter 1984), that is used in model selection. There is another method known a s Bayesian information criterion (BIC). By avoiding technical details, we shall only outline the results of these two methods when they are applied to the batch data set.

We set up a procedure for computing the relative merits of one model against another using these two methods. Both methods turn out to need optimisation. The results of the optimisation produces AIC= 552.8813 and BIC = 575.2849. We can use the relative values of the AIC and BIC statistics as guides in the model selection process. In this example, the AIC criterion favours the second model, while the BIC criterion favours the first model with fewer parameters. It is better to bear in mind here that since BIC imposes a greater penalty for additional parameters than does AIC, BIC always provides a model with a number of parameters no greater than that chosen by AIC [3].

3.11 Summary

We set out in this chapter to show that standard RBFs optimisation based on ordinary least squares is not suitable for modelling demand processes of the types considered in this thesis. Superficial examination of the approximation power of RBFs leads one to believe that the complexity of non–linear optimization in neural network has been transformed to linear regression, so that if the parameters are estimated consistently, the approach will provide not only the optimal estimates but means of doing model reduction. Closer scrutiny of the data sets, however, reveals that not only are the errors correlated, but also, because of higher orders non–linear dependence of the error sequences, our data sets are heteroskedastic.

We have found that using the results of non–linear transformation on the data sets, rather than the levels of our variables may be helpful. Like RBFs, such non–linear transformation, allows the explanatory variables to enter non–linearly in the regression equation. We were also able to show that , in the case of volatility, if one is working with long–range forecast horizons, the following asymptotic behaviours hold for the outputs of variance prediction:

[3]There is a reject curve in(Davis and Vinter 1984), showing the relationship between model efficiency and complexity. This reject curve motivates our study of model reduction using AIC and BIC.

1. The conditional standard deviation prediction approaches the unconditional standard deviation of ϵ_t given by the square root of formula (3.6.1). The nonlineal optimisation do not affect the minimum mean square error forecast of the conditional mean. The forecast approaches the unconditional mean of y_t as in the constant variance case. That is, the presence of volatility effects introduces dependence in the variance process, and only affects the uncertainty of the mean forecast, leaving the mean prediction itself unchanged.

2. The mean-square-error of the conditional mean approaches the unconditional variance of y_t.

Chapter 4

Optimization of covariance structures

4.1 Introduction

> One distinguishing idea that dominates many issues in optimization theory is convexity .... An important reason is the fact that when a convex function is minimised over a convex set, every locally minimal solution is global. Also, first order necessary conditions turn out to be sufficient. A variety of other problems conducive to computation and interpretation of solutions ride on convexity as well. In fact, the great watershed in optimisation isn't between linearity and non linearity, but between convexity and non convexity.
>
> (Rockafellar 1970)

This work represents the first attempt to demonstrate that ideas expressed some years ago by Rockafellar, which recently appeared remote from the confine of data analysis research, may now become a real medium for examining urgent problems of covariance structures. In chapters (2 and (3) our analysis problems were formulated to render the optimisation of the variance of the residuals as in chapter (2) or the innovation (martingale) sequences as in chapter(3). In chapter (2) the errors that entered the structural relations were assumed to be independent Gaussian and in chapter (3) were assumed martingale dependence, making the noise sequences correlated.

In this chapter we shall make no restriction of the errors but we still assume the

Markovian framework for the derivation of conditional expectation. In this chapter, we present the design in a state space form. In statistical terms, the formulation of problems in this chapter is much akin to Multivariate analysis, as opposed to univariate analysis in chapter (2).

In addition, the objective of model reduction in the last two chapters resulted in solving a nonlinear optimization problem. In this chapter the optimisation of the covariance structure results in optimization of a convex objective function over convex constraints. Apart from these essential differences, this chapter is the continuation of a singular spirit of doing data analysis on processes having Markov dependence characteristics using analysis of variances.

This chapter is organised as follows. The background that inspired us to follow this kind of optimisation technique is given in section (4.2). Section (4.3) summarises the optimisation problems that can be handled via LMI. Such problems include optimisation when the objective function is a determinant of a positive matrix, a method that has applications in model reduction when we are faced with the problem of choosing training data points so as to minimize the information matrix (the in stochastic control error covariance matrix). Some few sections dealing with various problems that can be formulated as our covariance structure follow this section. Several case studies are provided using tea data set, house sale data set, RBFs data and education test scores data.

4.2 Preliminaries

Our interest in the optimisation of covariance structures came from the interest in the study of the solution of some nonlinear optimisation problems arising from the study of RBFs, Neuro–fuzzy systems and nonlinear systems identification (Tankeh, Azvine and Mamdani 1997a, Tankeh and Mamdani 1997b, Tankeh and Mamdani 1997c). The results of this study is somewhat a continuation of an investigation on the efficiency of training neural networks using non–linear optimisation which was reported in (Tankeh 1989). It became apparent that the difficulty of training such networks arises from the non–convexity of the associated objective functions despite the non–linearity of the equation systems. As a result of the developments in convex optimisation with Linear Matrix Inequalities (LMIs), we were able to study how to formulate data analysis and nonlinear systems identification problems in terms of (LMIs).

In data analysis research one considers a data matrix $D = \{x_{\alpha i}\}$ of n observations on p response variables. In addition, we assume that each row of x is a p dimensional vector having a normal distribution with a covariance matrix $\sum$ that satisfies a positive definite matrix equation of the form

$$\sum = \Lambda\Omega\Lambda^T + \Psi. \tag{4.2.1}$$

In principle, the first term on the right hand side represents some *latent* factors that are contributing to the observations in the varieties and the last term represents the covariance matrix of the error of approximation.

The underlying basis of the modelling is that there is a set of *latent variables* f consisting of r underlying factors such that each observation vector x is some linear combinations of factors plus error. That is

$$x = \mu + \Lambda f + \epsilon,$$

where μ is the mean of x, Ω and Ψ are the covariance of f and ϵ, respectively. In reality, we are provided with the sample values of $\sum$ and we are interested in estimating Ω and Ψ such that (4.2.1) holds . The optimisation problem boils down to the following sub problems:

1. find positive matrices Ω and Ψ,

2. how many factors r are needed to provide a good approximation.

This mathematical model provides a multinormal structure that is richer to the extent that it is capable of explaining the *dependence* relationships that exist in the covariance structures. It can take account of the fact that there may be some natural ordering among the responses of the multivariate systems. For example, in claims and demand processes the sizes of the claims might be ordered by the times at which successive claims were reported. In any case it can be assumed that this time–orderings follow from *external* criteria that are impacting observations, thus regarding the data as coming from a *time series.* In this last context the data is recorded at the time scale t at particular points p. Similarly, other orderings can be introduced depending on the nature of the tasks, for example, in feature selection we might rank according to different structural scales.

The study of this covariance originated out of our involvement with the selection, weighing and comparing of the hidden neurons in radial basis function(RBFs). A typical RBF learning procedure consists of combining a number of activation of

hidden layer neurons to yield an overall index score when they are subjected to real training data. For example, (Chen, Billings and Grant 1992) obtained an index of neuron characteristics by combining output scores from each neurons. Thus, this provides an index of internal consistency of the RBF models that measures how adequately the composite score represents each of the neurons. In fact, Billings provides a reliability index for each neuron and rejects those neurons that do not account significantly in the variability of the observations. In our study of covariance structures, we have in mind a similar objective to Billings, in that we are also interested in model selection and ranking rules according to their relative merits. However, our approaches are different and the scope much wider. Nevertheless, Billings' work which is based on orthogonal least–squares is a special case of a much richer mathematical structure.

Although not exhaustive, the cases considered here suffice to show that there are substantial statistical and optimization problems that are related to (4.2.1). The study of (4.2.1) arises naturally under similar idealizations as we shall see in minimum trace factor analysis (Lawley 1956, 1967). A typical example being example (Bentler 1972), who obtained index of people status by combining rating scale of occupation, amount of income and source of income. Such an index provides a measure of internal consistency among the attributes.

The central problem here is that we do not have the exact distribution of the noise. Our first approach is to seek for a lower bound of the index of consistency. This approach amounts to characterising Ω, its properties under the conditions imposed by the output covariance $\sum$ and our prejudice about the nature of the noise in Ψ. Hence, it seems clear that another way of finding the internal consistency is seek for the upper bound error a matrix Ψ that is diagonal and satisfies

$$\begin{aligned} &\sum -\Psi > 0 \\ &\text{Diagonal } \Psi > 0. \end{aligned} \tag{4.2.2}$$

From the solution of (4.2.2), say $\acute{\Psi}$, we can obtain matrix of the internal state of the system Ω. Obviously, the approximation will be efficient if the coefficient of reliability

$$\rho = 1 - \frac{Tr\left(\acute{\Psi}\right)}{Tr(\sum)} \tag{4.2.3}$$

is high (Bentler 1972). The formulation of this optimisation is similar to other engineering design problems which assumes that in many real applications, the

standard approximation cannot adequately capture all the specifications. For instance, in H_∞ theory (Green and Limebeer 1995) noise attenuation are expressed in terms of worst–case scenarios. Similarly, Fletcher only allows the noisy term to take the worst possible value: since any other noise location will conform with a better internal approximation. This makes the result of such a LMI optimisation highly desirable in practice.

If $\acute{\Psi}$ is the maximum with respect to the constraint (4.2.1) then the problem is called an *educational testing problem of Fletcher.* It is named after Fletcher (Fletcher 1985), who first formulated this problem. A related problem to it was formulated in (Allwright 1988) which is a kind of *matrix approximation problem* motivated from Quasi–Newton methods. These problems which are optimisation problems with positive definite matrix constraints in which some or all of the elements of a matrix are variables have now been solved using LMI theory.

4.3 Semi–definite programming (SDP) with LMI

Although, the theory of semi–definite SDP is very highly developed, we shall limit attention here to examples of how it can be used for the optimisation of our covariance structure. The reader will find out, as we do, that one does not really need to know all the technical details of interior point algorithms[1], which is the method upon which the LMI optimization tool kits are based, to be able to use them. The applications of LMI shift the burden of the difficulties to that of problem specifications and the structuring of such specifications in terms of linear matrix inequalities.

Technical details about SDP can be found in (Apkarian and Gahinet 1995, Allwright 1989, Boyd, Ghaoui, Feron and Balakrishnan 1994, Karmarkar and Ramakrishnan 1991). The list of references is quite a lot and we have just named a few. The present study takes a different and much more *application–oriented* view, providing the straightforward perspective and simple geometrical insight that a data analyst needs.

The designation (LMI) has been applied to linear matrix of the form

$$L(\theta) = L_0 + \theta_1 L_1 + \ldots + \theta_n L_n > 0 \tag{4.3.1}$$

[1]More about interior point algorithms can be found in the user group home page http://www.mcs.anl.gov/otc/interiorPoint.

where

- $\theta = (\theta_1, \theta_2, \ldots, \theta_N)$ is a vector of unknown scalar, which is the optimization variables,
- $" >"$ implies "positive definite , i.e. the smallest eigen value of $F(\theta)$ is positive.

The entries of θ are often called the decision variables.

There are a few collections of generic SDP problems, three of which are:

1. The *strick feasibility problem:*

$$\text{Find} \quad \in \theta \in R^n \text{ such that} \quad L(\theta) > 0. \tag{4.3.2}$$

2. *Linear objective minimization* under LMI constraints:

$$\text{Minimise } C^t\theta \text{ subject to} \quad L(\theta) > 0. \tag{4.3.3}$$

3. *Determinant maximization* under LMI constraints:

$$\text{Maximise } \log\det L(\theta) \text{ subject to} \quad L(\theta) > 0, \ G(\theta) > 0, \tag{4.3.4}$$

where $G(\theta)$ defined as in (4.3) with constant symmetric matrices $G_0, G_1, \ldots, G_m$. All these three problems are convex. For example the objectives $c^T\theta$ and $\log\det L(\theta)$ are all convex functions on their respective constraint sets, which are themselves convex. Another generic problem that is important for data mining research is the generalized eigenvalue minimization problem discussed in (Boyd and Ghaou 1994).

4.3.1 Softwares and applications

There are several softwares tools for SDP. The LMI control Toolbox for Matlab (Mathworks 1995) provides a good user–interface that allows to specifiy the SDP with matrix stricture. In many problems the optimisation matrix has special structures. Adopting the notations from (Allwright 1988) and those in the introduction. In (Fletcher 1985) the structure of a positive definite matrix is

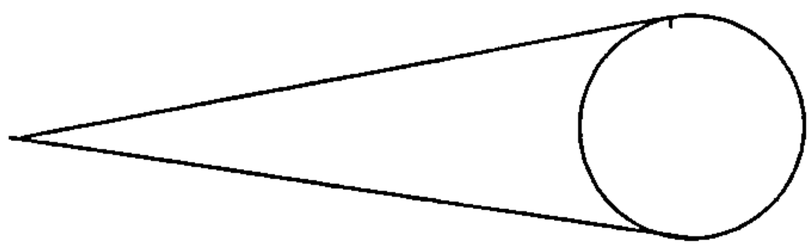


Figure 4.1: Positive definite matrix cone

defined as a closed convex cone of dimension $\frac{1}{2}n(n+1)$. The cone looks like figure (4.1), where, for example if $n = 2$ and

$$\begin{bmatrix} a & b \\ b & c \end{bmatrix},$$

then the cone $S^n_{\geq}$ is the convex region $\{(a,b,c) : a \geq 0, b \geq 0, ab \geq c^2\}$. Using geometric idea from linear algebra, it is known that any set $C \in R^n$ is a cone if $x \in C$ implies that $\lambda x \in C$ for each $\lambda > 0$, by assuming that the vextex of the cone lies in the origin (Rockafellar 1970). Therefore, for a matrix $X \in S^2_{\geq}$ the number of decision variables is 3. This variable can be vectorised and put into a single vector variable $x = (a,b,c)$, and we write $Vec(X) = (a,b,c)$. It is also possible to put any matrix variable X in the form

$$X(\theta) = L_0 + \theta_1 L_1 + \ldots + \theta_n L_n > 0$$

as we stated above. For example, the matrix $X \in R^{n \times n}$

$$\begin{bmatrix} 0 & 1 \\ -x & -y \end{bmatrix}, = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} + x \begin{bmatrix} 0 & 0 \\ -1 & 0 \end{bmatrix} + y \begin{bmatrix} 0 & 0 \\ 0 & -1 \end{bmatrix}.$$

The simplest and the most important LMI is the autoregressive system

$$\dot{y}(t) = Ay(t) \tag{4.3.5}$$

which is the equation that started off the history of LMI in 1890, when Lyapunov publised his work. He showed, using a P– quadratic lyapunov function that (4.3.1) is stable if $P \in S^n_{>}$ such that

$$A^T P + PA < 0. \tag{4.3.6}$$

We can appreciate the matrix stuctural format mentioned above if we check the operations involved in (4.3.6). There are multiplication operations on $n \times n$ matrices. Therefore the order of the operations is $O(n^3)$. But if we vectorise the

LMI then the LHS of (4.3.6) becomes

$$\sum_{i=1}^{n}\sum_{j=1} x_{ij}(A^T L_{ij} + L_{ij}A,$$

which has order of operations $O(n^4)$,

The optimisation process makes use of interior–point algorithms (Nemirovski and Nemirovski 1994)

4.3.2 Why SDP methods

The following reasons make SDP appealing:

1. A lot of problems in data analysis involve matrix and quadratic constraints, which can be expressed as LMI problems.

2. There are so many tools for solving SDP problems.

3. We can easily interpret the solutions of most SDP problems, sometimes through duality. It is very hard to explain solutions of neural networks or genetic algorithm solutions.

4. The problem of "regularisation" of solution in neural networks can easily be solve in SDP framework. One can ask in LMI toolbox that the solution x should lie in the ball

$$x^T x < r^2,$$

without increasing the complexity of the solution process.

4.4 Factor analysis

Our problem of studying *multivariable dependence* structure through the use of the covariance of the observations, can be understood in terms of factor analysis.

The important point in our covariance structure is the connection between it and factor analysis research. We shall begin from the parametric form

$$z_\alpha = \Lambda f_\alpha + \mu + \epsilon_\alpha \tag{4.4.1}$$

where z_α is the observation vector, f_α is a column vector of m components, representing some factors and μ is vector of p components mean of z_α. Let $q = p - m$. The coefficient matrix Λ is a $p \times m$ of rank $m < p$, called *factor loadings*. Our assumptions are

$$E[z] = \mu, \;\; E[\epsilon] = 0.$$

Let $\Omega = E\left[ff^T\right]$ which may be taken as the correlation matrix and $\Psi = E\left[\epsilon\epsilon^T\right]$ which may be diagonal. Then the covariance matrix of z becomes

$$\sum = \Lambda\Omega\Lambda^T + \Psi, \tag{4.4.2}$$

which is the form of equation (4.2.1).

This model is a factor analysis model if the conditions specified are observed. The central idea is that for a given set of observations one wants to find a set of intrinsic or latent factors $f_1, f_2, \ldots f_m$, fewer in numbers than the observed variates that will explain the intercorrelation of the response variates, in the sense that when the factors are singled out from the observed variates there is no longer any correlations between them.

Equation (4.4.1) can not be verified directly but (4.4.2) can be tested statistically if $(p-m)^2 - m < p$, reasons are given in (Maxwell and Lawley 1971).

4.5 Educational testing theory

For more on this test theory, see (Birenbaum, Kelly and Tatsuoka 1993) and (Lord and Novick 1968) who observed that there are many functions which can model test scores.

We assume that a test score x is a sum of a true score τ (a hidden trait) and an error score ϵ, where τ and ϵ are assumed to be uncorrelated. In a *congeneric* test it is assumed that in a set of test scores $x_1, x_2, \ldots, x_p$ with true scores $\tau_1, \tau_2, \ldots, \tau_p$ every pair of true scores τ_i and τ_j have unit correlation. Such a test can be represented as

$$x = \mu + \beta\tau + \epsilon, \tag{4.5.1}$$

where $x^T = (x_1, x_2, \ldots, x_p)$, $\beta^T = (\beta_1, \beta_2, \ldots, \beta_p)$ is the regression coefficients, $\epsilon^T = (\epsilon_1, \epsilon_2, \ldots, \epsilon_p)$ is the vector of error scores, μ is the mean vector of x and τ is the true score, for convenience scaled to zero mean and unit variance. x, τ and ϵ

are regarded a random vectors for a population of students being examined. Therefore the covariance matrix of x is

$$\sum = \beta\beta^T + \Psi^2, \tag{4.5.2}$$

where $\Lambda = \beta$, $\Omega = I$.

4.6 Fletcher problem

The study of this problem has a long history[2] . Although this problem has a number of important applications, the one that is of relevant to us is the *educational testing problem (Fletcher 1985):* that given a symmetric positive definite matrix $\sum$ how much can be subtracted from the diagonal of $\sum$ and still retain the positive semi-definite matrix. This problem has been tackled unsuccessfully. This problem can be formulated as a *primal* optimisation:

$$\begin{array}{ll} \text{minimise} & \text{trace}\Psi \\ \text{subject to} & \sum -\Psi \geq 0 \\ & \Psi \geq 0 \end{array} \tag{4.6.1}$$

Using the property of Schur complement, two constraints in (4.6.1) can be put together as a single constraint

$$\begin{bmatrix} \sum -\text{diag}(d_1, d_2, \ldots, d_n) & 0 \\ 0 & \text{diag}(d_1, d_2, \ldots, d_n) \end{bmatrix} \geq 0. \tag{4.6.2}$$

With recent development in SDP, this problem can be formulated as

$$\max \sum_{i=1}^{n} d_i \text{ such that } \begin{bmatrix} \sum -\text{diag}(d_1, d_2, \ldots, d_n) & 0 \\ 0 & \text{diag}(d_1, d_2, \ldots, d_n) \end{bmatrix} \geq 0. \tag{4.6.3}$$

4.7 Procrustes problem

There are several versions of Procrustes problem but we shall consider the *orthogonal Procrustes problem* . One finds an orthogonal matrix Q that rotates B

[2]We omit technical details here, including actual derivations of results.

into $A = BQ$ in such a way that the following nearness optimisation problem is solved

$$\text{Minimise } \|BQ - S\|_F, \text{ subject to } Q^TQ = I, \tag{4.7.1}$$

where F denote the Frobenious norm and $A \in R^{n\times n}$. To give the importance of the Procrustes problem, we shall choose an example much closer to home–in machine learning and pattern recognition. made some comments during a discussion on Procrustes Problem (Goodall 1991). They reported a test score experiment in which the student signs his name n times a special pad that records the pressure and coordinates of the pen 1000 times per second. They assumed a centred signature matrices Y_i, $i = 1, 2, \ldots, n$, each of dimension $N \times 2$ and representing n corresponding points form the leading S of signatures and consider the following optimisation

$$\text{Minimize } E = \sum_{i=1}^{n} \|Y_iA_i - S\|_2, \;\; S^tS = I, \tag{4.7.2}$$

over $S_{n\times 2}$ and A_i, a general 2×2 matrice. They obtained a simple solution

$$\hat{E} = \text{Minimize } \sum_{i=1}^{n} \|(Q_iQ_i^T - I)S\|_2 = ntrace\left(S^T(I - Q_iQ_i^T)S\right), \tag{4.7.3}$$

where A_i is replaced with an optimal estimate $\hat{A}_i = R_i^{-1}Q_i^TS$, computed from the QR–decomposition of Y_i, $Y_i = Q_iR_i$. The optimisation problem (4.7.3) is a well–known and can be solved through eigen value decomposition.

With no constraints on Q in the optimisation contained in (4.7.1) is the standard least–squares problem with solution $Q = B^+S$ where B^+ is pseudo–inverse of B.

4.8 Clustering with covariance matrix

Referring to the Educational testing problem, one is interested in the estimation of $B = \Lambda$, called the *factor loading matrix* in such a way that it will have a simple interpretation. The question that arises in education testing, factor analysis and food preference analysis is a model reduction one: how many factors f_i are needed to make a good approximation and which matrices $\hat{\Lambda}$ and $\hat{\Psi}$ give good approximations. This question gets very tricky because if U is an orthogonal matrix then $U\hat{\Lambda}$ is just as good as $\hat{\Lambda}$ since

$$\hat{\Lambda}^T\hat{\Lambda} = (U\hat{\Lambda})^T(U\hat{\Lambda}).$$

The diagonal matrix

$$W = \hat{\Lambda}^T \hat{\Lambda}$$

is very important and its diagonal elements are called *commonalities*. The total commonality is $trace(W)$. Using the property of trace, the we can see read the contribution made by each factor on the variance observed, which is similar to the problem of finding the effect of each neuron on the output or a RBFs or neural networks.

We shall show later that interpreting test results in terms of hidden factors introduces a large scope of subjective judgement,and as a consequence, different investigators might interpret the same data differently. This problem is also common in preference analysis of consumer goods, evaluation of house sale in different areas and other data analysis where the analyst seeks for hidden factors behind the observations. This kind of subjectivity in interpretation of results, however natural, is generally thought to be undesirable. To avoid this, scientists both in machine learning and statistics, have proposed various empirical techniques for transforming factors, which in some predetermined sense lead to a unique set of results.

Another way of stating this estimation problem is to say this: Give a matrix S of rank p find a matrix of smaller rank, say q that best approximates S. To see the application of this, suppose that we know the noise Ψ, then the estimation is

$$\hat{W} = \sum -\Psi,$$

where $\hat{W}$ is assumed to be of rank p. Suppose we are looking for only a limited number of factors $q = 2$ then we have a minimisation problem

$$\hat{E} = \text{Minimise} \sum_{i=1}^{n} \|W(\sum -\Psi)\|_F \tag{4.8.1}$$

where W is of rank $q < p$, $p \times q$ matrix of rank q. It can be shown that this optimisation can be solved using the theory of principal component, some combination of the work in(Allwright 1988) and (Davis and Vinter 1984). Note $x = (x_1, x_2, \ldots, x_p)^T$ the observation vector with covariance $\sum$. We wish to define a new coordinate system $w_1^T x, w_2^T x, \ldots, w_p^T x$. We need the new random variables $w_i^T x$ to be uncorrelated, which is essentially a principal component constraint. Then

$$cov\left[w_i^T x, w_j^T x\right], \quad i \neq j,$$

which provides a *sequential prediction of* x given $W^T x$. The error covariance is

$$E\left\{\left[x - E(x|W^T x)\right]^T \left[x - E(x|W^T x)\right]\right\}$$

which we saw in chapter (2) we need to minimise with respect to the with respect to w_1 subject to the orthogonality conditions. It turns out, using the theory of Schur complement in the appendix, that minimisation of the error covariance is the same as minimising the trace of the schur complement of the matrix

$$\sum = \begin{bmatrix} \sum & \sum w \\ w^T \sum & w^T \sum w \end{bmatrix}. \tag{4.8.2}$$

After repeating this process for all q its turns out that the optimal estimate of $\hat{W}$ the product of a diagonal matrix of the first q largest eigen values of $\sum$ and the matrix of those eigen values. We shall illustrate the application of this method to house evaluation and tea data sets.In fact (Allwright 1988)[3] obtained a related results without using conditional density and estimation theory.

The procrustes problem is also similar to a problem considered by (Anderson and olkin 1978). Here, one is given two given covariance matrices (C_1, C_2) and the objective is to construct random vectors distributed according to $X \sim (0, C_1)$ and $Y \sim (0, C_2)$ such that the l_2-distance between them is minimal. Consider the following matrix

$$\sum = \begin{bmatrix} C_1 & X \\ X^T & C_2 \end{bmatrix}. \tag{4.8.3}$$

We shall illustrate this covariance analysis with an example. Take the housing data for two groups(locations) with covariance matrices (C_1, C_2) and we are interested to know:

1. The largest possible correlation that might exist between the two groups.

2. The minimum correlation that might exist in the two groups.

Computing this, we have

$$C_1 = 1.0e+009 * \begin{bmatrix} 0.2737 & -0.1780 & 0.1539 & 0.3674 \\ -0.1780 & 1.3331 & -0.4461 & -0.8286 \\ 0.1539 & -0.4461 & 0.7201 & -0.1774 \\ 0.3674 & -0.8286 & -0.1774 & 4.2917 \end{bmatrix}$$

[3]Also see Woodgate PhD thesis,in control section, Imperial College.

$$C_2 = 1.0e+009 * \begin{bmatrix} 0.1936 & -0.0487 & -0.0198 & 0.6848 \\ -0.0487 & 0.8684 & -0.2140 & -0.1282 \\ -0.0198 & -0.2140 & 0.7935 & -0.1538 \\ 0.6848 & -0.1282 & -0.1538 & 3.8163 \end{bmatrix}$$

Using LMI toolbox the optimisation produces the estimate

$$\hat{X} = 1.000000e+09 * \begin{bmatrix} 0.0454, & -0.0846, & -0.0271, & 0.6147; \\ -0.0846, & 0.0179, & -0.0362, & -0.5802; \\ -0.0271, & -0.0362, & -0.1636, & -0.0228; \\ 0.6147, & -0.5802, & -0.0228, & 3.5081 \end{bmatrix}.$$

Look at

$$\hat{X}^T\hat{X} = 1.0e+019 * \begin{bmatrix} 0.0388 & -0.0361 & -0.0008 & 0.2234 \\ -0.0361 & 0.0345 & 0.0021 & -0.2097 \\ -0.0008 & 0.0021 & 0.0029 & -0.0072 \\ 0.2234 & -0.2097 & -0.0072 & 1.3022 \end{bmatrix}.$$

$$C_1C_2 = 1.0e+019 * \begin{bmatrix} 0.0310 & -0.0248 & 0.0098 & 0.1589 \\ -0.0658 & 0.1368 & -0.0508 & -0.3387 \\ -0.0084 & -0.0526 & 0.0691 & -0.0625 \\ 0.3054 & -0.1250 & -0.0631 & 1.6764 \end{bmatrix}.$$

Notice that $C_1C_2 = \hat{X}^T\hat{X}$ which shows that the optimal

$$\hat{X} = \sqrt{C_1C_2},$$

proved in (Anderson and olkin 1978). It is interesting to note that this problem has a very multivariate analysis of variance interpretations. The random variable $A - B$, representing the differences between the two groups have the following bounds on their covariance

$$\begin{aligned} \sum_{A-B} &= C_1 + C_2 - 2C_1C_2 \quad \text{lower} \\ \sum_{A-B} &= C_1 + C_2 + 2C_1C_2 \quad \text{upper.} \end{aligned}$$

If we are interesting in univariate case in which A and B are just scalar, say *land value in each of the neighbourhood,* then the bounds will be given by

$$\hat{X} = \sqrt{\sigma_1\sigma_2},$$

where σ_i, $i = 1, 2$ is the variance in each group. We can now carry out the same

procedures in the other groups and, from the results of the optimisation we can classify these group accordingly.

Using these covariances we need to know how related the two groups of scores are. To do this, we think about this as a clustering problem– that of looking for a distance measure. It turns out that this problem is equivalent to looking for a new random variable $Z = (A, B)$ such that the covariance of Z is $\sum$, given in (4.8.3) such that the l_2–distance between A and B are minimised. From SDP the expected l_2–distances between two matrices is

$$E\left[trace(A-B)^T(A-B)\right] = trace(C_1 + C_2 - 2X)$$

and this need to be minimised. The constraint $\sum \in S_>^n$ is the same as

$$C_1 - XC_2^{-1}X^T > 0.$$

Therefore the covariance optimisation is

$$\max trace 2X \text{ subject to } C_1 - XC_2^{-1}X^T > 0.$$

4.9 Optimal design of training data matrix A^TA

We have assumed that the non–linear RBF network $\hat{f}(.)$ is appropriate for approximating the system in the sense that the given data (yt, xt)'s satisfy the relations

$$E[y_t] = \hat{f}(x_t, \theta). \tag{4.9.1}$$

The optimal design for linear regression has been well developed but for non–linear RBF or neuro-fuzzy networks, not much is known. For linear regression, such as for the RBF weights λ, we consider N measurements $y_1, y_2, \ldots, y_N, x_1, x_2, \ldots, x_N$ of $\theta \in R^{p+1}$:

$$y_i = a_i\theta + \epsilon_i, \;\; i = 1, 2, \ldots, N \tag{4.9.2}$$

where a_i are constant row vectors in the notation used in section 2 and assume that the noise term ϵ_i is a normally distributed random variable with zero mean and covariance $\sigma^2 I$. In this notation, the least square estimator (2.2.6) may be written as:

$$\hat{\theta} = \left(\sum_{i=1}^{N} a_i^T a_i\right)^{-1} \sum_{i=1}^{N} a_i^T y_i, \tag{4.9.3}$$

and the error covariance P, defined as in (??) is given by:

$$P = E\left[(\theta - \hat{\theta})(\theta - \hat{\theta})^T\right] = \sigma^2 \left(\sum_{i=1}^{N} a_k^T a_i\right)^{-1}. \tag{4.9.4}$$

In case, the "experimental design problem" may be posed as: given test vectors τ_i, we need to choose $a_i \in \{\tau_1 \tau_2 \ldots \tau_M\}$ to make the error covariance P *small.* In general, this is a hard combinatorial problem since the optimal $\hat{P}$ will depend on the numbers $n_1, n_2, \ldots, n_M$ of each type of test performed. If we define the fraction of tests with $a_k = \tau_i$ as $\alpha_i = n_i/N$ then $\alpha_i \geq 0$ and $\sum_{i=1}^{M} \alpha_i = 1$, and the error covariance

$$P(\alpha) = \left(\sum_{i=1}^{N} a_i^T a_i\right)^{-1} = \frac{\sigma^2}{N}\left(\sum_{i=1}^{M} \alpha_i \tau_i^T \tau_i\right)^{-1}. \tag{4.9.5}$$

However, we have also to define what we mean by a small covariance P. The following criteria can be used:

1. **E–optimal design**

$$\min \lambda_{\max}\left(P(\alpha)\right) \tag{4.9.6}$$

2. **A–optimal design**

$$\min Trace\left(P(\alpha)\right) \tag{4.9.7}$$

3. **D–optimal design**

$$\min \det\left(P(\alpha)\right), \tag{4.9.8}$$

where $\lambda_{\max}(.)$ denotes the maximum eigen value of the matrix and $Det(.)$ denotes the determinant.

$P_j(\alpha)$ is the variance of the estimate θ_j which is proportional to the j^{th} diagonal element of $(A^T A)^{-1}$ and the covariance of θ_j and θ_k is proportional to the $(j,k)^{\text{th}}$ off–diagonal element. Since, in this section our interest is on the comparison of different design matrices, the value of σ^2 is not relevant.

Testing of hypotheses about the individual parameters θ_j may be carried out using the t test as mentioned earlier. However, if we wish to test for several parameters, the F test (see, for example (Atkinson and Donev 1992), whose notations we have followed) should be used and the $100(1-\alpha)$ per cent confidence region for all the $q = p + 1 = n_c + 1$ elements of θ is of the form:

$$(\theta - \hat{\theta})(A^T A)((\theta - \hat{\theta}) \leq q s^2 F_{q,\nu,\alpha} \tag{4.9.9}$$

where s^2 is an estimate of σ^2 on ν degrees of freedom and $F_{q,\nu,\alpha}$ is the α percentage point of the F distribution on q and ν degrees of freedom. The region defined by (4.9.9) is an ellipsoid with volume proportional to the determinant $\|A^TA\|$. Thus, the designs that maximise $\|A^TA\|$ are called D–optimum.

These are standard ideas in statistics but it is interesting to note that we may formulate these designs in terms of convex optimisation problems involving matrix inequalities, thus opening up the possibility of using the LMI solution techniques. For example, the D–optimal design may be formulated as:

$$\log\det\left(\sum_{k=1}^{M}\alpha_i\tau_i^T\tau_i\right)^{-1} \tag{4.9.10}$$

$$\text{subject to} \qquad \left(\sum_{k=1}^{M}\alpha_i\tau_i^T\tau_i\right)^{-1}, \tag{4.9.11}$$

$$\sum_{i=1}^{M}\alpha_i = 1, \alpha_i \geq 0, i = 1, 2, \ldots, M. \tag{4.9.12}$$

The "dual" approach (see Boyd 1994 for an explanation of what is meant by "dual" here) to solution of this design problem is thus based on seeking a minimum ellipsoid that contains the test data points. Such an ellipsoid can be obtained by solving a convex optimisation problem by maximising the log–determinant of a special covariance matrix. Figure (4.2) shows an example minimum ellipsoid centred on the origin that contains all the given test data points. The resulting matrix $\hat{P}$ of the error covariance may also be obtained from solution of the optimisation problem.

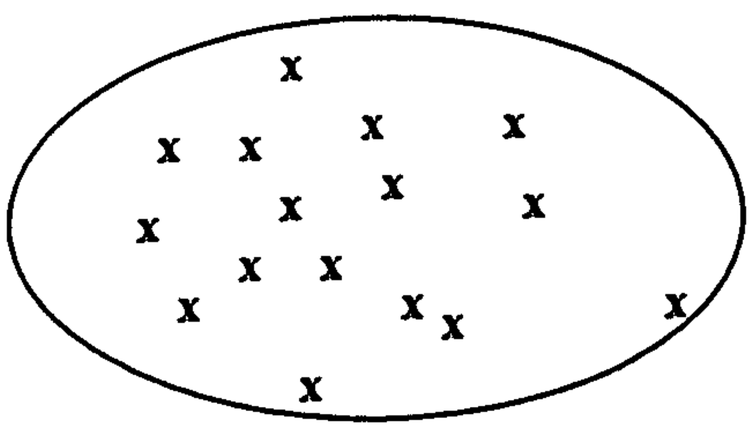


Figure 4.2: Clustering with minimum ellipsoid

4.9.1 Results using the tea data

Sensory Panel Attributes: $P, S_1, S_2, S_3, \ldots, S_8$. We computed the covariance, and for convenience, we have indicated just some few elements of the matrix

$$\sum = \begin{bmatrix} 3.8902 & 2.5990 & 0.0272 & 1.5722 & -2.0392 & 0.0726 & \ldots \\ 2.5990 & 3.5810 & -1.4713 & 3.0573 & -2.9412 & 2.1871 & \ldots \\ 0.0272 & -1.4713 & 1.7833 & -2.2821 & 1.0139 & 0.7819 & \ldots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \quad (4.9.13)$$

The eigen values
(16.7936, 9.0497, 1.7870, 1.6635, 0.4083, 0.2769, 0.1927, 0.0456, 0.0056). Fletcher problem seeks the solution to the minimisation

$$\text{Trace}(\Psi),$$

such that

$$\sum -\Psi > 0, \;\; \Psi > 0,$$

where Ψ is a diagonal matrix. Notice that this provides an optimisation with 9 decision variables and two linear matrix inequality constraints specifying their positiveness. We specify the following algorithm parameters:

- absolute tolerance $= 8.31124e-07$,
- relative tolerance $= 8.31124e-07$,
- a parameter called $NU = 10$, which controls the rate of convergence,
- the maximum iteration $= 100$.

The following results were obtained:

- Optimal solution was obtained after 18 iterations, and the optimization stopped because ABSOLUTE TOLERANCE was reached.
- Objective value obtained $= 0.531262$.

- $\Psi =$

$$\begin{bmatrix}
0.0008, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000 \\
0.0000, & 0.0668, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000 \\
0.0000, & 0.0000, & 0.0023, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000 \\
0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000 \\
0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000 \\
0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000 \\
0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.2871, & 0.0000, & 0.0000 \\
0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.1744, & 0.0000 \\
0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000, & 0.0000
\end{bmatrix}$$

The optimisation consists of two phases. In the first phase, the initial choose of the decision variable is not feasible, so such proceeds outside the feasibility region with the following primal objective function, dual objective function and duality gap at each stage in the iteration:

Primal objective	Dual objective	Duality gap
$1.00e-08$	$-1.68e+00$	$1.68e+00$
$4.47e-01$	$-1.68e+00$	$2.13e+00$
$3.09e-01$	$-2.33e-01$	$5.41e-01$
$4.05e-02$	$-7.36e-02$	$1.14e-01$
$1.28e-02$	$-2.78e-02$	$4.06e-02$
$4.82e-03$	$-1.48e-02$	$1.97e-02$
$2.33e-03$	$-9.81e-03$	$1.21e-02$
$-4.10e-04$	$-6.75e-03$	$6.34e-03$

The results of the second phase of the simulation are shown below:

$$
\begin{array}{ccc}
-3.57e-01 & -4.57e+04 & 4.57e+04 \\
-3.51e-01 & -4.57e+04 & 4.57e+04 \\
-3.27e-01- & 1.35e+04 & 1.35e+04 \\
-3.25e-01 & -7.16e+02 & 7.16e+02 \\
-3.27e-01 & -1.90e+02 & 1.90e+02 \\
-3.36e-01 & -1.33e+01 & 1.30e+01 \\
-3.71e-01 & -3.80e+00 & 3.43e+00 \\
-4.80e-01 & -8.99e-01 & 4.19e-01 \\
-5.08e-01 & -6.01e-01 & 9.32e-02 \\
-5.24e-01 & -5.52e-01 & 2.75e-02 \\
-5.29e-01 & -5.40e-01 & 1.03e-02 \\
-5.30e-01 & -5.34e-01 & 3.07e-03 \\
-5.31e-01 & -5.32e-01 & 1.03e-03 \\
-5.31e-01 & -5.32e-01 & 3.65e-04 \\
-5.31e-01 & -5.31e-01 & 1.25e-04 \\
-5.31e-01 & -5.31e-01 & 3.93e-05 \\
-5.31e-01 & -5.31e-01 & 8.53e-06 \\
-5.31e-01 & -5.31e-01 & 1.74e-06 \\
-5.31e-01 & -5.31e-01 & 2.66e-07
\end{array}
$$

Let the solution be $\acute{\Psi}$. We can then evaluate according to (4.2.3) the index of consistency or then the coefficient of reliability

$$\rho = 1 - \frac{Tr\left(\acute{\Psi}\right)}{Tr(\sum)} = 0.9824. \tag{4.9.14}$$

To interpret this result, it is important to notice that the internal structure Λ and Ω do not appear in the Fletcher's framework. This is contrary to the traditional study of data mining using RBF networks, where attention is focused on the hidden neurons, now represented by the regression matrices Λ and Ω. However, the result provides the bound of what can be expected of such internal structure. From a stochastic analytical view point, it tells us that in the worse possible case that those internal factors are highly consistent if ρ is high and less so if it is low. On a related matter one can derive consistency index from the correlation matrix, by rescaling $\sum$.

It is important to observe that the worst case eigen values of $(\sum -\Psi)$ are

$$(16.74, 9.00, 1.72, 1.54, 0.32, 0.00, 0.230.15, 0.20).$$

This is of rank 8 and if we neglect factors of very small eigen values, the rank will be even much smaller, say 6. Which is to say that there are six factors that are contributing to the results of the tea preference scores.

4.9.2 Results using test of English as a Foreign language

The matrix:

$$\sum = \begin{bmatrix} 94.7 & 87.3 & 63.9 & 58.4 \\ 87.3 & 212.0 & 138.7 & 128.2 \\ 63.9 & 138.7 & 160.5 & 109.8 \\ 58.4 & 128.2 & 109.8 & 115.8 \end{bmatrix} \tag{4.9.15}$$

is the covariance matrix for four different measures of English as a Foreign Language (Lord and Novick 1968). This covariance matrix is based on the sample of 1416 students. In (Bentler 1972) the same covariance matrix $\sum$ was used and our aim here is to reproduced these results using LMI optimization. Using a different optimization method, Bentler obtained a diagonal matrix Ω of elements

$$(47.58, 36.98, 42.76, 13.29).$$

Our results give

$$(47.57, 36.99, 42.77, 13.28).$$

He also found that the hidden factors should be the worst case eigen values

$$(427.42, 14.97, 0.00, 0.00),$$

is obtained by computing the eigen values of $(\sum -\Psi)$. We also obtained a set of worse case eigen values

$$(427.42, 14.97, 0.00, 0.00).$$

It is important to note that, using LMI optimisation, we have been able to reproduce Bentley's results. It is important to observe that in both approaches the rank of $(\sum -\Psi)$ turns out to be 2 and the index of reliability of the test using my own scheme turns out to be 0.7588.

4.9.3 Preference analysis with tea demand sample

In the tea data sets the following sensory Panel Attributes are provided for 20 types of tea:

Code	Description
S_1	Colour Of Leaf
S_2	COLOUR: Other
S_3	COLOUR: Uniformity Of Colour
S_4	COLOUR: Fibre
S_5	SHAPE/SIZE: Perceived Lea Size
S_6	SHAPE/SIZE: Dust
S_7	SHAPE/SIZE: Uniformity Of Leaf Size
S_8	SHAPE/SIZE: Other

P	S_1	S_2	...	S_8
3.329	6.050	2.560	...	...
3.700	5.659	2.577	...	...
4.004	5.442	1.495	...	...
...	...	...	...	

We shall illustrate the idea behind identification of linear relations in noisy data using the tea sample with only two attributes. In this case $p = 2$, each observation point $(y_{1\alpha}, y_{2\alpha}^T)$ is a 2–vector distributed as

$$N\left[\begin{pmatrix} z_{1\alpha} \\ z_{2\alpha} \end{pmatrix}, \sigma^2 I_2\right], \quad \alpha = 1, 2, \ldots, n, \tag{4.9.16}$$

where the learning parts satisfy the linear equation

$$z_{2\alpha} = \gamma + \beta z_{1\alpha}, \quad \alpha = 1, 2, \ldots, n, \tag{4.9.17}$$

where γ and β are to be determined. The model is illustrated by figure (4.3). The point $P(y_{1\alpha}, y_{2,\alpha})$ is a typical observed point, with other points indicated similarly, and point $B(z_{1\alpha}, z_{2\alpha})$ is the point whose estimate $(\hat{z}_{1\alpha}, \hat{z}_{2\alpha})$ we wish to compute in such a way that the optimal line OA is perpendicular to PB. Thus, the parameters of the learning process is taken to be $z_{11} z_{12} \ldots z_{1n}, \gamma, \beta$, and σ^2. We can determine $z_{21} z_{22} \ldots z_{2n}$ using likelihood function. Therefore, the likelihood estimates are also least squares estimates[4]. There, estimates are the generalized least squares in the sense that it minimizes that sum of squares of distances of the observed points

[4]Note that the estimates are the same whether σ^2 is known or not.

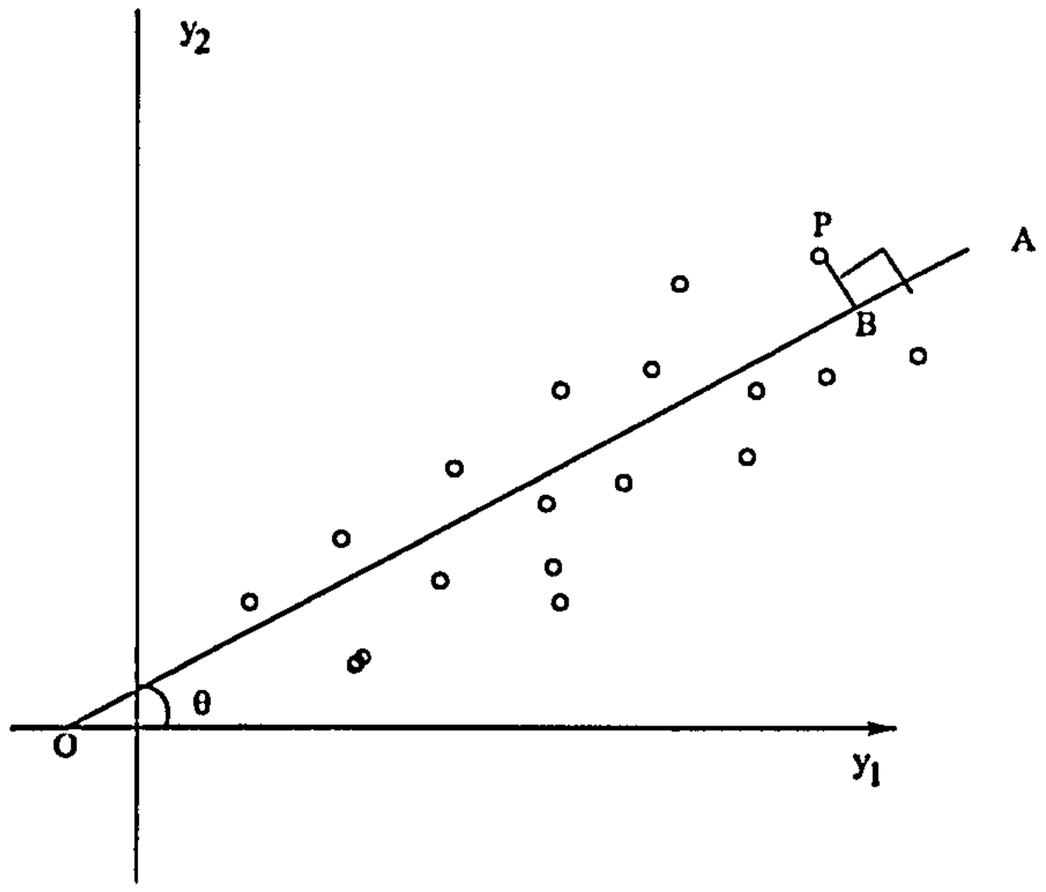


Figure 4.3: Relationship in tea preference for $p = 2$

from the fitted line. Furthermore, the estimated points $(\hat{z}_{1\alpha}, \hat{z}_{2\alpha})$ are the projections of the observed points are the line.

With respect to the principal component analysis, the direction characterized by the line is the only direction (unique) which:

1. lies on the first principal component,
2. maximises the scatter within groups as in classification,
3. has projections of greatest variance.

Further geometric analogue can be developed in the space of dimensionally equal to the number of observations n. Consider the sets of the points

$$v_1 = (z_{11} - \bar{z}_1, \ldots, z_{1n} - \bar{z}_1), \;\; v_2 = (z_{21} - \bar{z}_2, \ldots, z_{2n} - \bar{z}_2), \;\; . \tag{4.9.18}$$

From (4.9.17), it implies that the two points are collinear

$$v_2 = \beta v_1 \tag{4.9.19}$$

and the random vectors of the observations

$$y_1^* = (y_{11} - \bar{y}_1, \ldots, y_{1n} - \bar{y}_1), \;\; y_2^* = (y_{21} - \bar{y}_2, \ldots, y_{2n} - \bar{y}_2) \tag{4.9.20}$$

have independent spherical normal distributions with centres at v_1 and v_2,

respectively. Furthermore, point v_i is a on the hyperplane defined by

$$\sum_{\alpha=1}^{n} \{z_{i\alpha} - \bar{z}_i\} = 0, \quad i = 1, 2.$$

These two $(n-1)$-dimensional hyperplanes intersect on the line defined in (4.9.19). Thus, the support vector points are the solution of the following quadratic optimisation problem

$$\min \|y_1^* - v_1\| + \|y_2^* - v_2\| \text{ subject to } \begin{cases} v_2 = \beta v_1 \\ v_1 \in \text{ hyperplane } \langle e^T, v_1 \rangle = 0 \\ v_2 \in \text{ hyperplane } \langle e^T, v_2 \rangle = 0, \end{cases} \quad (4.9.21)$$

where e is a vector with elements of ones.

4.9.4 Identification from noisy tea demand

Now, let us consider the whole tea data attributes $(S_8, S_7, S_6, S_5, S_4, S_1, S_2, P, S_3)$. To illustrate the application of this method to the tea preference data we use the correlation between the 9 measurements for a sample 20 tea types. The covariant matrix C, is computed. The characteristics roots and vectors are given below.

Assume that the covariance of f_α is I and ϵ_α are I and ψ, respectively. Then

$$\sum = \Lambda\Lambda + \psi, \quad (4.9.22)$$

is the covariant of equation (4.4.1). For simplicity consider only one case of noise perturbations: $\psi = \sigma^2$. Here the sensory scores (attributes) are uncorrelated and have equal variances. The variability of S_i and S_j are the same and have the same standard deviation. It is important to note that the covariant structures of ν_α satisfy the linear relationship

$$B\Theta = 0. \quad (4.9.23)$$

where $\Theta = \Lambda\Lambda$ and as we have seen, the maximum likelihood estimate $\hat{B}$ is

$$B = V_{II}^T. \quad (4.9.24)$$

From (Lawley 1967)

$$\hat{\Theta} = V_I D_I V_I^T + \hat{\sigma}^2 V_I V_I^T \quad (4.9.25)$$

$$\hat{\Theta} = V_I D_I V_I^T + \hat{\sigma}^2 V_{II} V_{II}^T \quad (4.9.26)$$

Suppose that the n independent observations have been taken on a p–dimensional random variable with distribution $N(\mu, \sum)$.

With respect to the geometric analysis, the following interpretation is of interest. If the scores x of the teas are normally distributed, then the contours of equal density are ellipsoids, of the form $(x-\mu)^T \sum^{-1}(x-\mu) = 1$. This motivates the following quadratic optimisation problem:

$$\min d = (x-\mu)^T(x-\mu) \text{ subject to } \left\{ (x-\mu)^T \sum^{-1}(x-\mu) = 1 \right. , \tag{4.9.27}$$

Let $\bar{x} = (1/n)\sum_{\alpha=1}^{n} x_\alpha$ and

$$C = \frac{1}{n}\sum_{\alpha=1}^{n}(x_\alpha - \bar{x})(x_\alpha - \bar{x})^T, \tag{4.9.28}$$

These are the likelihood estimates:

$$\hat{B} = V_{II}^T, \tag{4.9.29}$$

where

$$V_{II} = \begin{bmatrix} S_2 & P & S_3 \\ 0.1573 & -0.6329 & 0.0306 \\ -0.3473 & 0.4714 & -0.0127 \\ 0.5567 & 0.4316 & -0.0846 \\ 0.6005 & 0.2136 & 0.2155 \\ 0.0479 & 0.1795 & 0.1685 \\ -0.2696 & 0.3231 & -0.0851 \\ -0.2062 & 0.0867 & -0.0534 \\ -0.1986 & 0.0267 & -0.1134 \\ -0.1651 & 0.0219 & 0.9455 \end{bmatrix},$$

$$\hat{\nu} = V_I V_I^T(y_\alpha - \bar{y}), \ \alpha = 1, 2, \ldots, n, \tag{4.9.30}$$

where

$$V_I^T = \begin{bmatrix} S_8 & S_7 & S_6 & S_5 & S_4 & S_1 \\ 0.4215 & -0.2546 & 0.3509 & -0.0699 & 0.3006 & -0.3360 \\ 0.4160 & 0.1661 & -0.0933 & -0.4211 & 0.2143 & -0.4737 \\ -0.1105 & -0.3700 & -0.1398 & 0.2963 & 0.4228 & -0.2478 \\ 0.3412 & 0.4521 & 0.3217 & -0.1830 & -0.0765 & 0.2895 \\ -0.3887 & -0.1580 & 0.5596 & -0.1186 & -0.4604 & -0.4710 \\ 0.4613 & -0.4977 & 0.3451 & 0.2744 & -0.1731 & 0.3617 \\ -0.3953 & 0.0030 & 0.4633 & -0.2838 & 0.6196 & 0.3342 \\ 0.0225 & 0.5363 & 0.2951 & 0.7032 & 0.1572 & -0.2292 \\ -0.0046 & -0.0778 & -0.1055 & 0.1789 & 0.1688 & 0.0243 \end{bmatrix} \tag{4.9.31}$$

$$\hat{\sigma}^2 = \sum_{i=m+1}^{p} \lambda_i. \tag{4.9.32}$$

Observe the following:

1. The directions indicated by the column V_{II} are the directions of minimum scatter,

2. The directions V_{II} also minimises the volume of the ellipsoid of the quadratic form circumscribed by the covariance matrix C.

Assume that the covariance of f_α is I and ϵ_α are I and ψ, respectively. Then

$$\sum = \Lambda\Lambda + \psi \tag{4.9.33}$$

is the computed covariance with eigen values

$$(16.7936, 9.0497, 1.7870, 1.6635, 0.4083, 0.2769, 0.1927, 0.0456, 0.0056).$$

The eigen values of the covariance matrix C are ordered as $\lambda_1 > \lambda_2 > \ldots > \lambda_p$. Let the corresponding characteristic vectors be $V = [V_I, V_{II}]$, $V_I = [v_1, v_2, \ldots, v_m]$, $V_{II} = [v_{m+1}, v_{m+2}, \ldots, v_p]$, normalised by $V^T V = I$, where $m = 6$. If we write the entire data set for the tea experiment as

$$Y = \begin{bmatrix} \\ y_1^T \\ y_2^T \\ \cdots \\ y_n^T \end{bmatrix}.$$

The point for principle component is data reduction in the sense that there is some form of dimensionally reduction. Notice that the smallest k ordered eigen values of C, $\lambda_{m+1}, \lambda_{m+2}, \ldots, \lambda_p$ are small for some m, say 4. In this way any random observation y can be predicted well by $V_I^T y$.

What is important for model reduction is how much information is lost by using m principal components.

Given any observation y_α, we project it in the span of the vectors $v_1, v_2, \ldots, v_m$ using V_I, i.e. y_α becomes $V_I y_\alpha$. Thus, the expected value of y_α, given $V_I^T y_\alpha$:

$$E\left\{y|V_I^T y_\alpha\right\} = \bar{y} + CV_I(V_I^T CV_I)^{-1}V_I^T(y_\alpha - \bar{y}) \tag{4.9.34}$$

Therefore, for each y_α we need the residual

$$\hat{\epsilon}_\alpha = y_\alpha - E\left\{y|V_\alpha^T y_\alpha\right\} \tag{4.9.35}$$

and summing

$$\hat{\epsilon}^T\hat{\epsilon} = \sum_{\alpha=1}^{n}\left[y_\alpha - E\left\{y|V_I^T y_\alpha\right\}\right]^T\left[y_\alpha - E\left\{y|V_I^T y_\alpha\right\}\right] \tag{4.9.36}$$

It turns out that the prediction sum of squares errors is

$$\hat{\epsilon} = (n-1)\sum_{i=m+1}^{q}\lambda_i. \tag{4.9.37}$$

The proof of these facts are tedious, but a careful look at (Anderson and Moore 1979, Davis and Vinter 1985) will find such accounts behind Kalman filtering informative.

The maximum possible prediction error occurs when we use no principal components at all, i.e.

$$\hat{\epsilon}_{\max} = \sum_{i=1}^{n}(y_i - \bar{y})^T(y_i - \bar{y}) \tag{4.9.38}$$

$$= tr\left\{\sum_{i=1}^{n}(y_i - \bar{y})(y_i - \bar{y})^T\right\} \tag{4.9.39}$$

$$= (n-1)tr\,(C) \tag{4.9.40}$$

Thus, the value

$$\rho = \frac{100\sum_{i=m+1}^{n}\lambda_i}{\sum_{i=1}^{n}\lambda_i}.$$

is the percentage of maximum error left unexplained by projecting the observation data $y_1, y_2, \ldots, y_n$ on the span of the first m principal components. Using the tea preference scores $\rho = 24.5\%$

There are two important reasons why we use principal components to analyse data for tea consumption research:

1. If the number of sensory factors q is very large then the analysis of these factors may be untenable. Thus, we must find means of reducing the dimensionally.

2. The precise factors that make people demand one type of tea more than the other is so complicated that the decomposition of our study to 8 sensory scores $(S_1, S_2, \ldots, S_8)$ against the preference P make us not to trust some of the information provided. Thus, we expect to have errors in the design matrix and it will be necessary to ignore directions corresponding to small eigen values.

The data reduction method based on this is closely related to the idea of identifying factors that determine the observed preference score.

However, it should be kept in mind that principal components are based on linear combinations of observations and if the important structure in the data is nonlinear, the method can totally miss the structure of the underlying relationship.

4.10 Factor analysis on tea demand

Data were collected on five variables $(x_1, x_1, x_2, x_3, x_4, x_5)$ on each house that were going on sale in different places. This example is the extension on the tea preference analysis[5] and we have replaced the 9×20 data with a vector of five variables representing;

- A letter form A, B, C, D, E, F, G.

- x_1 = Appraised land value.

[5]We actually used factor analysis here but in this part of the investigation we are restricted by Unilever from publishing the results of the total length of the tea data of dimension $9 \times 20 \times 500$, although we have been permitted to use the mean average summary of limited dimension 9×20 which we have used throughout out this work

- x_2 = Appraised value of improvement on the home.
- x_3 = Total appraised value of the home.
- x_4 = Sale price.

The test scores of house sales were conducted in 7 different neighborhoods A, B, C, D, E, F, G. The number of houses examined in the seven neighborhoods were $(42, 319, 258, 467, 500, 372, 482)$, a total of 2440 data points, i.e. 42 houses were examined in A, 319 examined in B, and so on.

This is similar to the Uniliver tea data set, where several hundred people were asked about their preference of twenty different types of tea, preference values collected on 9 variables $P, S_1, S_2, \ldots, S_8$. Our work on preference analysis is to investigate the pattern of relationships among the variables in an effort to understand better the dimensions underlying the demand for tea in particular and that of consumer goods in general.

Similarly, we can take the different neighbourhoods to represent tea types and the set of variables $\{x\}$ to represent the test scores, similar to the variables (1) $P, S_1, S_2, \ldots, S_8$ in tea study, (2) different subject scores in Fletcher's education testing, (3) different θ parameters for RBFs in model reduction in chapter (2). We computed the correlation matrix C as

$$C = \left\{ \begin{array}{ccccc} & x_1 & x_2 & x_3 & x_4 \\ x_1 & 1.0000 & 0.3045 & 0.2090 & 0.7931 \\ x_2 & 0.3045 & 1.0000 & 0.3765 & 0.4473 \\ x_3 & 0.2090 & 0.3765 & 1.0000 & 0.2314 \\ x_4 & 0.7931 & 0.4473 & 0.2314 & 1.0000 \end{array} \right\}. \tag{4.10.1}$$

Because there are four variables in the home sale data, there are four eigen values $0.6036, 0.9792, 0.1896, 2.2276$ with their corresponding eigen vectors.

Our goal of factor analysis is to summarise a pattern of correlations with as few factors as possible. Here we shall retain only the eigen values 0.9792 and 2.2276 which are the two largest. Their matrix of eigenvectors is

$$V = \left[\begin{array}{cc} -0.4339 & 0.5581 \\ 0.4079 & 0.4620 \\ 0.7291 & 0.3487 \\ -0.3373 & 0.5946 \end{array} \right]^T.$$

Note that $V^TV = I$ because they are orthogonal. Observe that

$$\begin{aligned} L &= V^TCV \\ &= \begin{bmatrix} 0.9792 & -0.0000 \\ -0.0000 & 2.2276 \end{bmatrix}, \end{aligned}$$

where C is given in (4.10.1). Therefore, with all the eigen values put together in the matrix L and the eigen vectors put together in the matrix U we have

$$C = ULU = U\sqrt{L}\sqrt{L}U^T = W * W^T.$$

W is called matrix of weights. Using this matrix as transformation operator (Maxwell Lawley 1971, Anderson 1963) the variates can be written in terms of $x = Wz$.

$$W = \begin{bmatrix} -0.1985 & -0.4294 & 0.2872 & 0.8329 \\ 0.5965 & 0.4036 & 0.0762 & 0.6895 \\ -0.4563 & 0.7215 & -0.0187 & 0.5205 \\ -0.0095 & -0.3338 & -0.3178 & 0.8874 \end{bmatrix}.$$

Thus, the linear transformation is applied to the p observed variates $x_1, x_2, x_3, \ldots, x_p$ in order to produce a set of uncorrelated and standardised variates $z_1, z_2, z_2, \ldots, z_p$. The matrix W is called the matrix of loadings and

$$x_1 = -0.1985z_1 + -0.4294z_2 + 0.2872z_3 + 0.8329z_4.$$

$$\ldots$$

$$x_4 = 0.0095z_1 + -0.3338z_2 + -0.3178z_3 + 0.8874z_4.$$

The sum of the roots is 4, so the percentage of variance explained by the

Variable	Name	Percent of variance explained
x_1	Appraised land value	15.1
x_2	Appraised value of improvement on the home	24.5
x_3	Total appraised value of the home	4.7
x_4	Sale price	55.6

If we use only the first two largest eigen values then the loading matrix is

$$W_1 = \begin{bmatrix} -0.4294 & 0.8329 \\ 0.4036 & 0.6895 \\ 0.7215 & 0.5205 \\ -0.3338 & 0.8874 \end{bmatrix}$$

which is obtained from $W_1 = V * \sqrt{L}$, where L is given above.

variable	Name	factor 1	factor 2	Comm.
x_1	Appraised land value	−0.4294	8329	87.81
x_2	Appraised value of ...	0.4036	0.6895	63.84
x_3	Total appraised value of the home	0.7215	0.5205	79.14
x_4	Sale price	−0.3338	0.8874	89.9

The factor loading matrix W_1 gives the correlation between the factors and the variables. The first component column of the matrix gives the correlations between the first factor and each variable in turn, Appraised land value(−0.4294), Appraised value of improvement on the home(0.4036), Total appraised value of the home(0.7215), Sale price(−0.3338). The second component or column is the correlation between the second factor and each variable in turn.

To interpret the factor it is important to check which variables it is highly correlated with. Thus, the second factor is primarily a wealth condition. The first factor is an added values to a home, something that is there, recent, new and transitional. Note that for the second factor the variables "Appraised land value " and "Sale price" are counterproductive having negative loadings. Inference is very obvious. From this, the communality, that is the sum of the squares loading for variable across factors, is the fraction of the observation of the variable that is due to the factors. For example, the 87.81% of Appraised land value is accounted for by factor 1 and factor 2. Usually a factor is most interpretable when a few variables are highly correlated with it and the rest are not. Another method of interpreting the data of the matrix loading, that is both informative and simple, is to plot the matrix points in the diagram in which the orthogonal axes are made to represent the factors and the coordinates of the points are the factor loadings. This is done in figure (4.4) and it is seen that the points appear to belong to two clusters that subtend an acute angle at the origin. This is supporting the hypothesis that there are two *latent* factors that are causing the observations.

$$Q = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}. \tag{4.10.2}$$

Now, using the transformation matrix Q in (4.10.2) we can transform the loadings W_1 into a new loading defined as $\vec{W}_1 = WQ$, where $\theta = 30\,\text{deg}$. This rotates the diagram in figure (4.4) into a new diagram in figure ((4.5), where loadings are all in the first quadrant and much easier to interpret. This is known as simple structure according to Louis Thurstone's interpretability criteria for factor

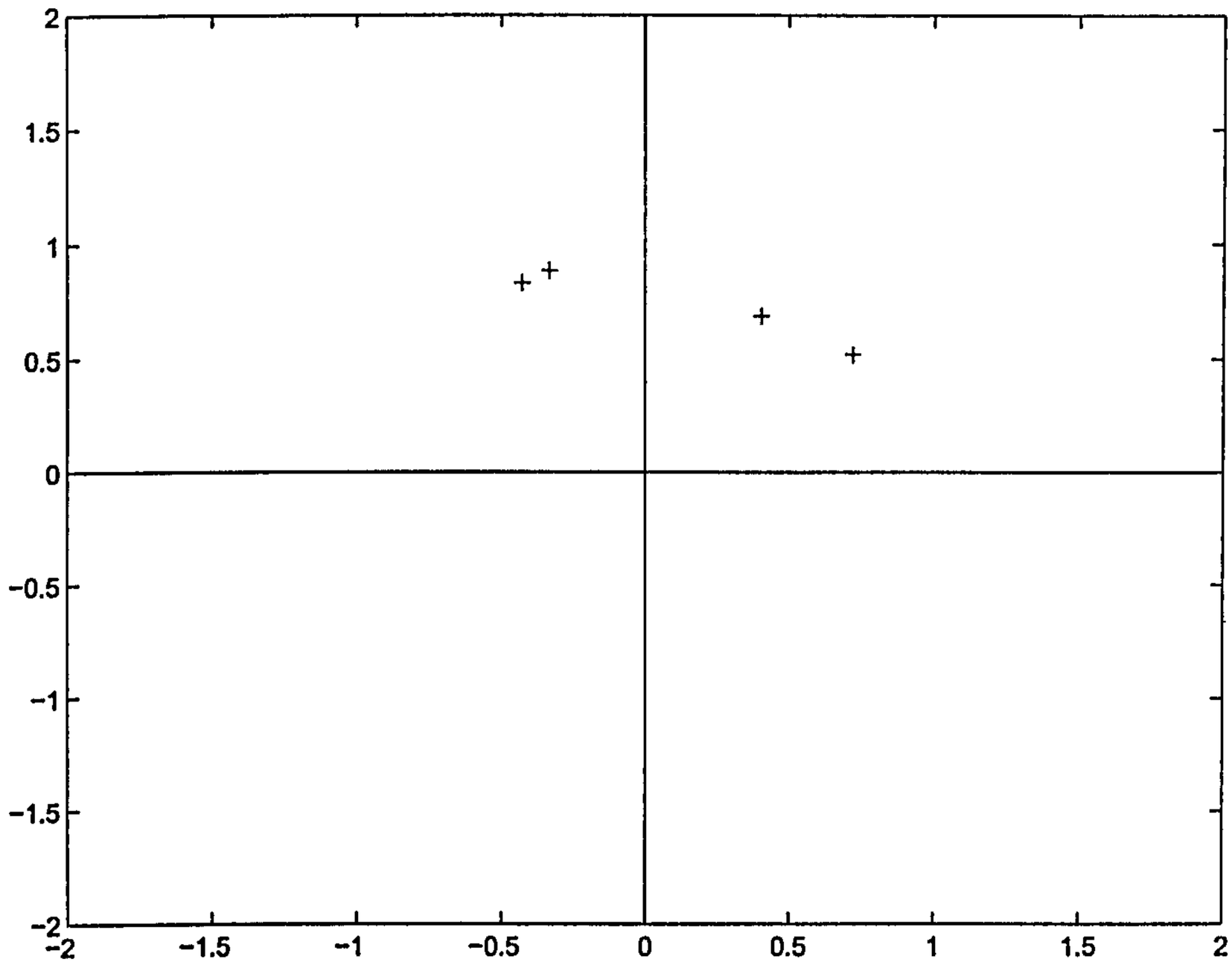


Figure 4.4: Planar view of factor loadings

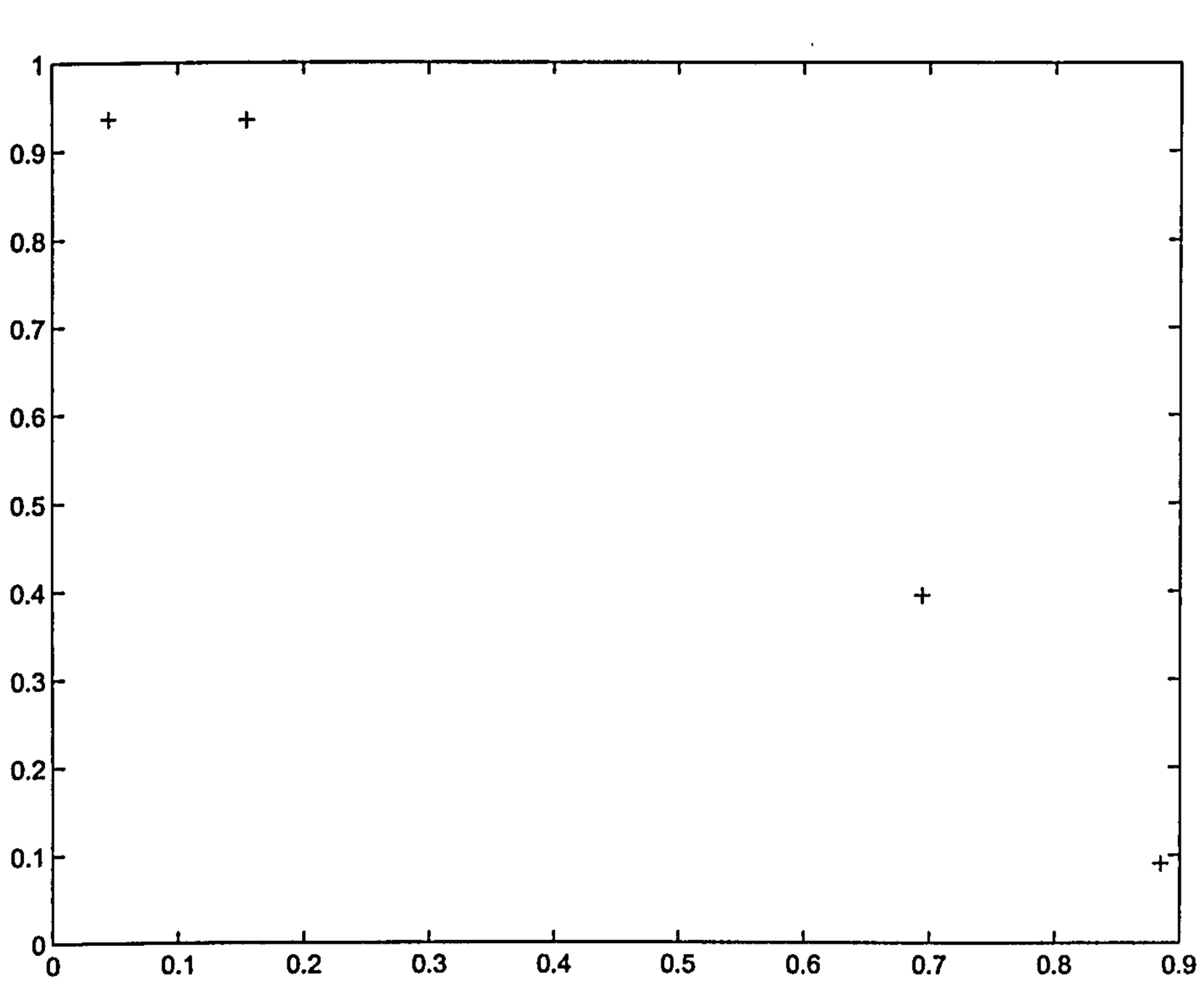


Figure 4.5: Rotation of factor loadings

structures (Maxwell and Lawley 1971). A factor matrix for k factors exhibits simple structure if

1. Each variable has at least one zero loading.
2. Each factor in a factor matrix with k columns should have k zero loadings.
3. Each pair of columns in a factor matrix should have several variables loading on one factor but not the other.
4. Each pair of columns should have a large proportion of variables with zero loadings in both columns.
5. Each pair of columns should only have a small proportion of variables with non zero loadings in both columns.

We have used graphs to find the transformation matrix. Using an optimization method described in (Maxwell and Lawley 1971), we obtained a loading matrix

$$Q = \begin{bmatrix} 0.828 & -0.561 \\ 0.561 & 0.828 \end{bmatrix}, \tag{4.10.3}$$

from an optimisation that converges after few iterations. We have avoided outlining the algorithms in used(interested reader should consult the reference for details).

4.11 Summary

This chapter takes a state space approach to the solution of the problem set out in the introduction. It takes the view that if there is any sense that can be made of the data we have provided, it can best be done by assuming an additive structure and our observation is composed of two parts: the true part (or signal part or latent part or hidden part) and the noise part. By making the observations into a vector of p components, we are able construct a Lyapunov–type linear matrix equation that can be optimised in various ways.

We are able to spot that this linear matrix equation is related to some factor analytical problems with similar structure like the education testing problem of Fletcher. At the optimisation end of the investigation, the problem was welcomed by Fletcher, but within the narrow framework limits its complete solution.

However, the release of Matlab LMI optimisation toolbox (Mathworks 1995) does provide us with a better vantage point than in Fletcher's days and we have provided ample examples to show this. Another method of solving this problem is to use principal component analysis as we have done.

The operative significance of these optimisation methods to preference analysis in demand and supply systems are demonstrated. Recall that in preference analysis we are interested in finding out what factors influence customers to buy one type of tea instead of others. In addition, we show thatr preference analysis is similar to house sale problem–factors that make one house to be more preferable by buyers than other houses in similar or different neigbourhoods. This is also similar to model reduction in RBFs, where one seeks some hidden neurons more than the others on the basis of how well they explain the observations.

In optimisation using principal component analysis on all the variates, we transform the attributes in such a way that the resulting variables are linearly uncorrelated. We show that unlike principal component, the aim of factor analysis is to predict the linear relationship of observed attributes in terms of much smaller numbers of hypothetical factors. The key result is whether there is correlation between one tea attribute, say uniformity of colour tea demand and the rest of the attributes that were tested by customers. From these studies, we were able to use the method of optimising covariance structures to obtain results that have been very useful to our industrial partners.

Chapter 5

Markov models for queues

5.1 Introduction

There is no doubt that Renewal theory provides the unifying tools for studying many stochastic processes. This possibly explains why several research monographs and books introduce renewal theory first: (Heyman and Sobel 1982, Kohlas 1982), Ross 1983 and Wolff 1989).

The study of Martingale calculus has found its greatest successes in control, first for diffusions, then for jump processes. It has also found its way into some of the classical areas of queueing processes, an account of this is given in the book by (Bremaud 1981). There is no doubt that PDP processes go a step further in its attempt to incorporate all non–diffusion models of applied probability in one framework.

Despite the richness of the PDP theory in providing an excellent framework for modelling most systems, it is now generally accepted that the amount of genuine applications of the theory is disappointedly small. It is equally well admitted that the general problem centres around the sort of integro–differential equations that arise when studying the generators of these processes. The generators of PDPs are poorly understood. There does not seem to be a unifying framework that links these generators. Each generator is specific to the application domain upon which it is derived and a slight change in boundary conditions give rise to a completely new generator.

Exceptions to this state of affairs are stochastic processes within the framework of diffusion processes, where the great successes that has been attained stochastic

control theory has been attributed to the fact that the problems reduce to investigating partial differential equations for which there is already a large body of knowledge.

Research in PDP, it must be admitted, pays too much attention to broad questions about existence of optimal controls, etc. and too little to solution or approximation of specific problems. In this chapter the goal is the application of renewal theory to the computation of distributions and expectations of functionals of a PDP process. As we shall see, there is a duality relationship between distributions and expectations of functionals of PDP. Both can be identified by the generator of the process, in that if we can compute the expectation $E_x[f(x_t)]$ for a measurable function $f \in B(E)$ then setting $f = I_A$ for measurable sets A results in the distribution $P_x\{x_t \in A\} = E_x[f(x_t)]$. The solution to any of these performance measures rests on the integro–differential formulae derived in Theorems (5.3.1) and (5.3.2).

Theorem (5.3.2) also provides us with a computational tool for calculating expected costs in renewal processes. For example, we can compute the expected number of renewals M_t in the finite interval $[0, t]$ by judiciously tuning parameters in Theorem (5.3.2),though this is not perhaps obvious. One important product of this paradigm is that it provides a suitable framework for relating the renewal integral equations (Cox 1962) and generators of PDPs. These generators provide opportunities for assessing conditions under which a process is a martingale, opening a big window of unexploited resources for renewal processes. We now turn to these processes.

The theory of renewal processes is a very elegant piece of mathematics; it has a number of applications, notably in the modern queueing theory. The classic books like (Asmussen 1987) in this area contain a great deal of material and may prove daunting for someone meeting the subject for the first time. This chapter provides a short and fairly gentle exposition of the relevant ingredients necessary to develop the important link between the renewal theory and generator of PDP. Only the main theorems are presented.

The most promising queueing models for broadband and multi–media applications in recent years are the $(M/D/1/N, M/G/1/N, G/G/1/N)$ queues. The simplicity of the first two models is due to the fact that the arriving process is Poisson and, therefore, Markov. However, this benefit is overshadowed with the fact that the departure processes are not Markov. For the purpose of control, the complexity of the dynamical structure, combined with the non–Markovian nature of the service

process, precludes any direct analytical approach and forces one to study various types of approximation techniques such as the theory of large deviations. The synthesis of various stochastic models such as the theory of large deviation have been proposed for the modelling and control of such queueing communication systems (Duffield 95a, Duffield 95b).

This work demonstrates how such queueing systems can be transformed into Markov processes. The approach is based on the application of a family of Markov processes known as *piecewise deterministic processes* (PDPs) (Davis 1993). The path of a PDP enjoys a deterministic dynamics punctuated by random jumps. In the special case of communication systems these jumps are generated when calls arrive (or depart), or when the buffering system is full. PDP models provide methods for computing expectations of performance measures such as mean queue length, cell loss probabilities, waiting time distributions, etc. These quality–of–service parameters are obtained from the solutions of integro–differential equations arising from the generators of PDPs. New systematic methods for the solutions of such equations have been established.

This work captures the dynamical structure of these systems through the infinitesimal generator of a semigroup. A functional is then defined in terms of this semigroup with a terminal cost penalty ϕ when the process hits a boundary time T. *This is why the problem is sometimes termed an exit–time control problem.* Through this formalism, the expected queue length at any time t can be evaluated from the solution of a system of first order partial differential equations with non–local interaction. A finite difference method is formulated for solving this system of equations.

The first rigorous developments of this modeling framework used the $M/G/1/N$ as an illustrative example. In an $M/G/1/N$ queue calls arrive as a Poisson process with rate λ. The service time $\{S_n\}$ has a general distribution G, and G is continuous and has a hazard rate function $\mu(t)$. The buffer has capacity $N-1$. This is a single server queueing system, where calls arrive from infinite sources into a single node with one channel with mean holding time $(1/\mu)$. The first call that arrives into an empty system goes into service immediately. Let T_n be the inter–arrival time of the customers. Hence, T_n is the time between the arrival of customer number $n-1$ and customer number n. Let x_t be the number of calls in the system at time t. Because the process $\{A_t\}$ is Poisson its inter–arrival times $\{T_n\}$ are independent and exponentially distributed. The "memoryless " property of the Poisson process ensures that $\{A_t\}$ is a Markov process. Hence, this explains the attachment of M in the queue model $M/G/1$. However, the process $\{x_t\}$ is not

a Markov process.

Similarly, let S_n denotes the service time of call number n. G in the notation relates to the fact that we do not know the distribution of the service process $\{S_n\}$. It is important to observe here that, even though we know the mean of G its distribution is still unspecified. Through this work it is assumed that $E(T_1), E(S_1) < \infty$ and, unless it is stated otherwise we shall write S for S_1.

The most promising queueing models for broadband and multi–media applications in recent years are the $(M/D/1/N, M/G/1/N, G/G/1/N)$ queues. The simplicity of the first two models is due to the fact that the arriving process is Poisson and, therefore, Markov. However, this benefit is overshadowed with the fact that the departure processes are not Markov. For the purpose of control, the complexity of the dynamical structure, combined with the non–Markovian nature of the service process, precludes any direct analytical approach and forces one to study various types of approximation techniques such as the theory of large deviations. The synthesis of various stochastic models such as the theory of large deviation have been proposed for the modelling and control of such queueing communication systems (Duffield 95a, Duffield 95b).

5.2 Other methods of modeling M/G/1/N queues

The steady state analysis for $M/G/1/\infty$ queue is well–known and we shall not like to spend much time on the topic. However, we are going to make a point that will be emphasized through this thesis- that because of the general non–exponential service time, the probability of a service–completion in any short interval $(t, T + \Delta t)$ is not independent of the past. Infact, [1] it depends on the so–called *residual service time* (time since the current service stated. It is also known as elapsed service time and in this thesis it will be denoted as τ.

It turns out that there are different ways of handling the complication introduced by the non–Markovian nature of for $M/G/1/\infty$ queue:

1. From renewal theory define a 2–dimensional stochastic process $\{(N_t, \tau)\}$ in which the state (N_t, τ) contains the relevant bit of past history. This kind of

[1] These ideas have been emphasized by Dr. Howl(private communication) and others in Imperial College.

process is a type of Markov process known as *piecewise–deterministic process (PDP).* This thesis is concerned with the application of PDP theory to non–Markovian queues like $M/G/1/N$.

2. One approach is to consider the so–called *departure times*–track down the values of $\{N_t\}$ at the instances of departures. This formation produces the sequence of values $\{N_i : i = 1, 2, \ldots\}$, where N_i is the number of items left behind at the i^{th} departure instant. This process forms the so–called *discrete time Markov chain,* $\{N_i\}$. It is not hard to see that if t_i stands for the i^{th} departure instant then the only events that occur between successive departures $[t_i, t_{i+1}]$ are arrivals. And the later are Poissons. and, therefore Markov. In addition, several standard literature on $M/G/1$ system refers to $\{N_i\}$ as an *embedded Markov chain* . It is also argued that the equilibrium state distribution of the continuous–time process $\{N_t\}$ is that of the embedded chain $\{N_i\}$. This is the motivation for analysis the later in order to infer information needed.

3. Another approach is known as *mean–value analysis.* It is the most common approach and it is very highly documented. One reason for its popularity is its simplicity and the fact that it can be applied to priority queueing systems.

4. The fourth approach is *computer simulation.* The behaviour of the system $M/G/1$ as it evolves over time is studied by developing a simulation model. In practice this model takes a set of assumptions about the components that form the system and how they operate.

5.2.1 Features of PDPs

The simplest PDP is a Continuous Time Markov Chain (CTMC). A class of Markov processes which can be used to model a great number of queueing communication processes are called *piecewise deterministic Markov processes* (PDPs). PDPs were initiated by Davis in (Davis 1984) and further developed into a rich theory in his book (Davis 1993).

The technical features of a PDPs theory can be found in (Davis 93). Here, we shall restrict ourselves to the generator of equation as follows. Let C be a Banach space of all bounded and uniformly continuous functions

on R^n, endowed with the sup–norm. Consider a PDP $\{x_t\}$ with sample path as in

(Davis 1993) then the generator A is defined as

$$Af(x) = \Xi f(x) + \lambda(x) \int_{y \in E} [f(y) - f(x)] Q(x, dy). \tag{5.2.1}$$

where

1. A is an integro–differential operator.
2. E is the state space of the process $\{x_t\}$.
3. $f(x)$ is the cost of being in state x.
4. $\Xi f(x)$ is a differential operator that describes the dynamics of the system if no jumps occur. For example in an $M/G/1/N$ system if no calls arrive then the amount of work load x_t is decreasing and follows a deterministic trajectory determined by the rate at which the server is processing the jobs.
5. $\lambda(x)$ is the rate at which jumps occur, i.e. $[\lambda(x)]^{-1}$ is **the mean sojourn time** in state x, i.e. the time interval between entry into state x and exit from x. Hence, $\lambda(x)$ is the **jump rate** at state x. It is also known as the hazard rate.
6. When a jumps takes place, the **post–jump location,** y, depends probabilistically on its starting point, x. $Q(x, dy)$ denotes the **jump distributions,** defined by

$$Q(x, dy) = P\{y \leq x_t < y + dy | x_{t-} = x\}, \tag{5.2.2}$$

   x_{t-} and x_t are the values of the process immediately before and immediately after a jump occurring at time t.
7. $Af(x)$, the expected rate of change of the cost $f(x)$, is the infinitesimal generator of a C_0 strongly continuous Markov semigroup on the Banach space C.

To introduce the principle behind PDPs, consider the point events starting at $t = 0$ occurring at times $T_1, T_2, \ldots$, where T_n denotes the time of the n^{th} event. For a call admission communication system T_n represents the arrival of the n^{th} call.

The trajectory of a PDP follows a deterministic flow until the first jump time. This trajectory can be described by a differential equation. The process continues until the flow hits the boundary ∂E of the state space E.

A PDP is a Markov process, with random jumps like continuous time Markov chains (CTMC). However, it is different from CTMC in the sense that the process does not necessarily stay constant between jumps: instead, the process $\{x_t\}$ follows a deterministic flow. The sample path of a PDP could look like the graph in figure (5.1).

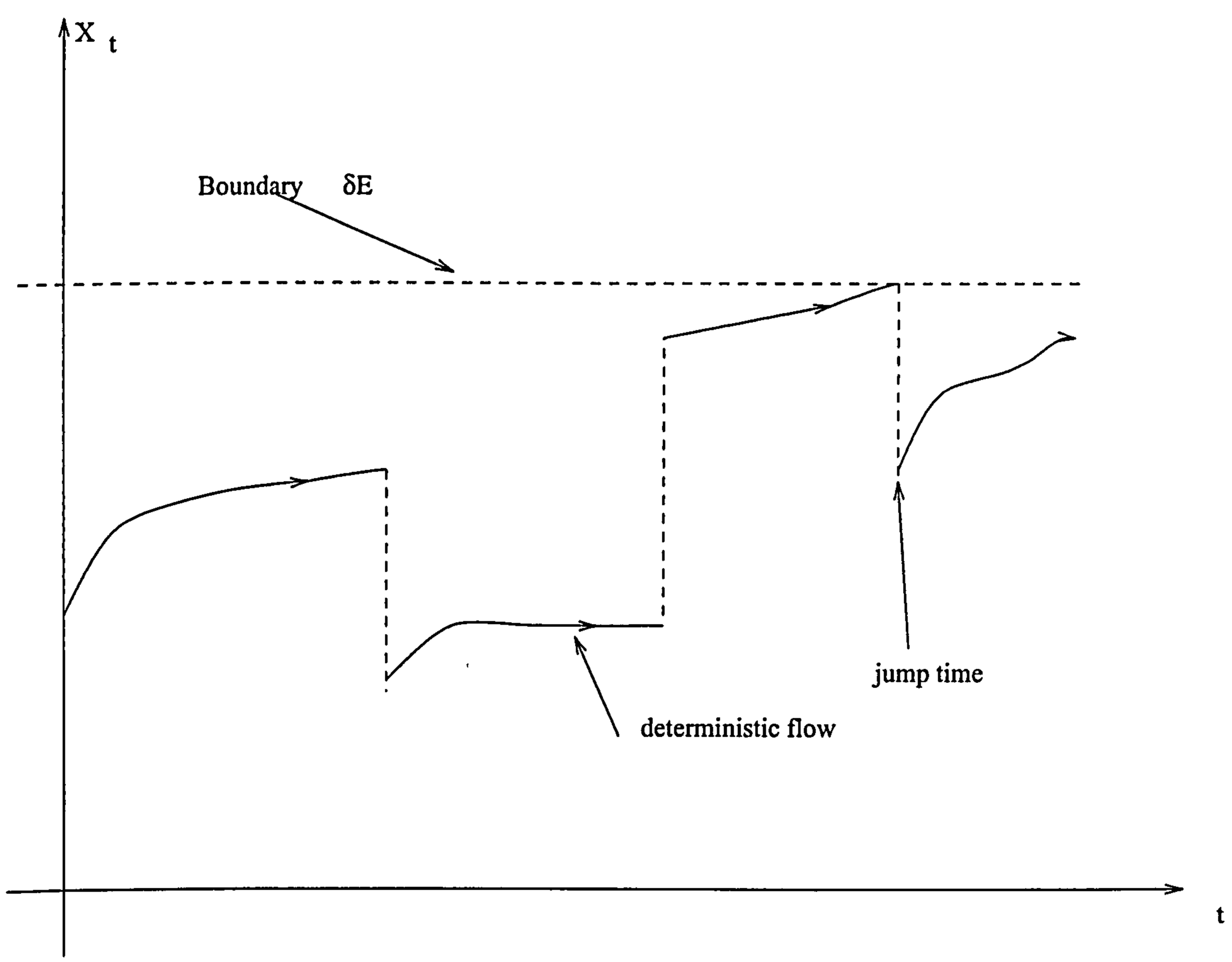


Figure 5.1: The sample path of a PDP process

A PDP is a Markov process with its deterministic trajectory in E that is punctuated with random jumps. Three essential features specify the process: the generator of the deterministic components, the jump rate and transition probability measure that account for the random component of the process.

Thus a PDP has the following features:

1. The deterministic part of the trajectory of $\{x_t\}$ can be specified by a vector field $g : E \to R^d$. g is a Lipschitz continuous function which determines a set

of integral curves ϕ on E, such that $\phi(t,x)$ is a solution to the ordinary differential equation (ODE)

$$\frac{d}{dt}\phi(t,x) = g(\phi(t,x)), \;\; \phi(0,x) = x \in E. \tag{5.2.3}$$

In (Davis 1993 page 52) Davis considers a differentiable function f and let $x(t) = \phi(t,x)$. Then

$$\frac{d}{dt}f(t,x(t)) = \sum_i \frac{\partial f}{\partial x^i}(x(t))g^i(t,x(t)), \tag{5.2.4}$$

where g^i is the i^{th} component of g. Here the state x is a d–dimensional vector, with i^{th} component x^i. Then the deterministic part of the trajectory of $\{x_t\}$ is given as

$$\frac{d}{dt}x^i(t) = g^i(t,x), i = 1,2,\ldots,d$$

We can write the equation 5.2.4 as

$$\frac{d}{dt}f(x(t)) = \Xi f(t,x(t)), \tag{5.2.5}$$

where Ξ is a first–order differential operator

$$\Xi = \sum_i g^i(t,x(t))\frac{\partial}{\partial x^i}, \tag{5.2.6}$$

To build up the model of the PDP we must first of all define the first exit times, $t^*(x)$, from the set E. Then the first exit time $t^*(x)$ is

$$t^*(x) := \inf\{t > 0; \phi(t,x) \in \partial E\} \tag{5.2.7}$$

for every $x \in E$. $t^*(x)$ is ∞ if the process never hits the boundary.

2. A jump rate $\lambda(x) : E \to R_+$ such that for each $x \in E$ there is a $\epsilon > 0$ such that

$$\int_0^\epsilon \lambda(\phi(t,x))dt < \infty, \tag{5.2.8}$$

the process does not manifest jump explosions.

3. A transition measure $Q(.;.) : E \cup \Gamma^* \to [0,1]$, where

$$\Gamma^* := \{z > \partial E : \exists t > 0, \exists x \in E \;\; s.t. \;\; z = \phi(t,x)\} \tag{5.2.9}$$

denotes the set of *active boundary* which the flow may reach (Dempster *et al*

1992). Observe that on reaching Γ^* the process $\{x_t\}$ necessarily jumps back to E.

The jump times can be characterised by a generalised negative exponential distribution. Starting at $x \in E$ we can represent the jump times as follows. Define $x_t = \phi(t, x)$ for $0 \leq t \leq \tau_1$, where τ_1 is the realisation of *the first jump time.* Jumps occurs at random times $0 < \tau_1 < \tau_2 < \ldots$ and τ_1 has distribution determined by

$$P_x(\tau_1 > t) = \begin{cases} \exp(-\Lambda(x)), & t < t^*(x) \\ 0 & t \geq t^* \end{cases}$$

where $\Lambda(x) = \int_0^t \lambda(\phi(s, x)ds$. When τ_1 is realised (possibly at $\tau_1 = t^*(x)$), we have $x_{\tau-} = \phi(t, x)$ and the *post jump state* x_{τ_1} has distribution given by

$$P_x\{x_{\tau_1} \in A|\tau_1\} = Q(A; \phi(\tau_1, x))$$

on the Borel sets A of E. The process restarts at x_{τ_1} according to the same recipe. The jump times satisfy the following relations:

$$\tau_i = \inf\{t > \tau_{i-1} : x_t \neq x_{t-}, \} \tag{5.2.10}$$

$$\tau_0 = 0, i \geq 1. \tag{5.2.11}$$

There are further conditions:

(a) **Boundary condition:** This relates to the jump which occurs when the process hits the boundary ∂E of the state space E. It is a necessary condition of the theory that $f(x)$ remains invariant, i.e.

$$f(x) = \int_{y \in E} f(y)Q(x, dy), \forall x \in \partial E. \tag{5.2.12}$$

This means that, on average, the value of $f(x)$ immediately after the boundary jump is the same as that just before the jump.

(b) **Bounded jumps:** We assume that the following expectation is bounded:

$$E\left\{\sum_{\tau_i \leq t} |f(x_{\tau_i}) - f(x_{\tau_i-})|\right\} \tag{5.2.13}$$

This condition implies that jumps must not be too large.

5.2.2 Generators of PDPs

It is now obvious that the generator of a PDP must have two components:

1. the generator due to the deterministic motion,
   and
2. the generator due to the jumps in $\{X_t\}$.

From equation (5.2.6) we know the generator of the deterministic component. Since the jump component of a PDP is a continuous the generator of the stochastic compoent just becomes

$$Af(x) = \int_{y\in E} \lambda(x)Q(x,dy)f(y) - \lambda(x)f(x) \tag{5.2.14}$$

$$= \lambda(x)\int_{y\in E}[f(y)-f(x)]Q(x,dy). \tag{5.2.15}$$

Equation (5.2.15) gives the expected rate of change due to jumps in $\{X_t\}$.

Combining equations (5.2.5) and (5.2.15) the generator of a PDP has the following form:

$$Af(x) = \Xi f(x) + \lambda(x)\int_{y\in E}[f(y)-f(x)]Q(x,dy). \tag{5.2.16}$$

5.3 Expectations of PDPs

5.3.1 Infinite–time horizon

For an infinite–time horizon we consider the cost functional

$$V(x) = E_x\left\{\int_0^\infty f(x_t)dt\right\}, \tag{5.3.1}$$

where $f : E \to R_+$.

We can think of $f(x)$ as the rate at which cost are incurred when $x_t = x$, so that V, defined by (5.3.1) gives the expected total cost incurred in $[0,\infty)$ starting from $x_0 = x$. Dynkin's formula states that if $V(x)$ is in the domain of the generator A then

$$E_x[V(x_t)] = V(x) + E_x\int_0^t AV(x_s)ds. \tag{5.3.2}$$

By setting $t \to \infty$ in (5.3.2), we obtain

$$\lim_{t\to\infty} E_x\left[V(x_t)\right] = V(x) + E_x \int_0^\infty AV(x_s)ds, \tag{5.3.3}$$

$$\lim_{t\to\infty} E_x\left[V(x_t)\right] = E_x\left\{\int_0^\infty f(x_t)dt\right\} + E_x \int_0^\infty AV(x_s)ds. \tag{5.3.4}$$

Theorem 5.3.1 *If $V(x)$ satisfies the condition*

$$\lim_{t\to\infty} E_x\left\{V(x_t)\right\} \to 0 \tag{5.3.5}$$

then $V(x)$ is the unique solution to the equation

$$AV(x) + f(x) = 0, \quad x \in E, \tag{5.3.6}$$

with the boundary condition

$$BV(x) = 0, \quad x \in \partial E, \tag{5.3.7}$$

where

$$BV(x) = \int_{y\in E} V(y)Q(x,dy) - V(x). \tag{5.3.8}$$

This is because (5.3.4) now becomes zero which implies (5.3.6). Turn to the finite time horizon problem.

5.3.2 Finite horizon expectation

Consider

$$V(x,t) = E_{x,t}\left\{\int_t^T f(x_\tau,\tau)d\tau + \Omega(x_T)\right\}, \tag{5.3.9}$$

where $\Omega(x_T) : E \to R_+$ is a bounded non–negative functional. Here the expectation $E_{x,t}$ is taken over the finite time interval $[t,T]$, starting at $x_t = x$. To solve this problem we transform the state space into the space–time process $\bar{x}_t = (x,t)$ for $t \in [t,T)$.

The new process $\bar{x}_t$ is now defined on the state space $(E \times [t,T]) \cup \{\Delta\}$. Δ is the usual cemetery state which is defined in such a way that the $\bar{x}_t = \Delta$, for $t \geq T$.

With these changes the expectation equation (5.3.9) now becomes

$$V(x,t) = E_{x,t}\left\{\int_0^\infty f(x_\tau,\tau)d\tau)\right\}, \tag{5.3.10}$$

where $f(x,T) = \Omega(x)$ for $x \in E$. The generator of the new process $\{\bar{x}_t\}$ is now given as

$$\bar{A}V(x,t) = \frac{\partial V}{\partial t}(x,t) + AV(x,t), \tag{5.3.11}$$

where A is the generator of the process $\{x_t\}$.

Theorem 5.3.2 *If V is in the domain of the generator, i.e. $V \in D(\bar{A})$, then it satisfies the equation*

$$\bar{A}V(x,t) + f(x,t) = 0, \ , \ x \in E \tag{5.3.12}$$

with the boundary conditions

1.

$$V(x,T) = \Omega(x), \ x \in E, \tag{5.3.13}$$

2.

$$BV(x) = 0, \ , \ x \in \partial E, \tag{5.3.14}$$

where

$$BV(x) = \int_{y\in E} V(y)Q(x,dy) - V(x), \ . \tag{5.3.15}$$

for all $x \in \partial E$.

In Theorem (5.3.2) the condition $\lim_{t\to\infty} E\,[V(\bar{x}_t] = 0$ is satisfied automatically because $\bar{x}_t = \triangle$ for all $t \to \infty$. The transition measure of the process is now defined in such a way that $Q(x,T,dy) = \sigma_\triangle(dy)$ and the jump rate $\lambda(\triangle) = 0$. σ_z is the Dirac measure at the cemetery state z and the transition is defined on the boundary $E \times \{T\}$. Using equation 5.3.11 we can write equation 5.3.12 in the form

$$\frac{\partial V}{\partial t}(x,t) + AV(x,t) + f(x,t) = 0, \ V(x,T) = \Omega(x) \tag{5.3.16}$$

which has a solution (5.3.9).

The following theorem is a corollary to theorem (5.3.2)

Theorem 5.3.3 *Suppose that instead E is an open set in R^d with boundary ∂E*

and

$$AV(x) = 0, \ x \in E, \tag{5.3.17}$$
$$V(x) = \Omega(x), \ x \in \partial E \tag{5.3.18}$$
$$\tag{5.3.19}$$

then

$$V(x) = E_x \left[\Omega(x_\tau\right], \tag{5.3.20}$$

where

$$\tau = \inf\{t : x_t \in \partial E\}.$$

Proof: Apply theorem (5.3.2) with $t = 0$ and $T = \tau$ the solution (5.3.9) is (5.3.20).

It is important to see the connection between theorems (5.3.1), (5.3.2) and diffusion processes. Although the PDP model is not a diffusion process they share a lot in common. To see this connection observe that

$$E_{x,t}\left[V(x_T, T)\right] = V(x,t) + E_{x,t}\int_t^T \left(\frac{dV}{ds} + AV\right)(x_s, s)ds \tag{5.3.21}$$

is the finite–horizon version of the Dynkin–formula. And $U_t V\bar{x} = E_{x,t}\left[V(x_T, T)\right]$ is a semigroup operator on $B(E)$.

In terms of probability theory (5.3.21) is equivalent to saying that

$$C_T^V = V(x_T, T) - V(x_t, t) - \int_t^T \left(\frac{dV}{ds} + AV\right)(x_s, s)ds \tag{5.3.22}$$

is a martingale. The result of Theorem (5.3.3) can also be obtained by applying Doop's optional sampling theorem from the martingale C_T^V of (5.3.22).

Another connection that is of greatest important to optimisation, optimal control and optimal stopping problems is that functionals of the form (5.3.1) and (5.3.20) are cost functionals that need to be optimised. As we shall see in the next chapter these functionals are evaluated by solving integro–differential equations.

In most application one is provided with the discounting problem with cost functional

$$V(x) = E_x \int_0^\infty e^{-\lambda s} x_s ds, \tag{5.3.23}$$

where λ is interpreted as a discount factor. Using the above property the generator

of the discounted process is $\lambda I - A$.

In fact a class of operators known as semigroup operators can be introduced as a general framework of studying Markov processes. The abstract theory of linear semigroup is then developed and applied to PDPs. Their properties can be studied, from which the essential theorems can be developed.

5.4 Generators of queueing processes

Regard the service (departure) process $\{d_t\}$ as a renewal process with inter–occurrences random variables

$X_1, X_2, \ldots$, that have density $g(y)$, i.e. X_1 is the service time of the first job, X_2 is the service time of the second job, etc. Denote the hazard rate correspond to renewal density by

$$\lambda(t) = \frac{g(t)}{\int_t^\infty g(s)ds}. \tag{5.4.1}$$

The denominator in (5.4.1) is the survival function. The renewal process $\{d_t\}$ is not a Markov process because $g(y)$ is not necessarily exponential distributed.

To make this process a Markov process we create a two–dimensional process $x_t = (d_t, \tau_t)$, where τ_t is the time since the last

departure. This method was first introduced by (Cox 1962, 1965). The

process $\{x_t\}$ is now a Markov process. In fact, it is

a PDP because the process will move on a deterministic trajectory at a unit speed if no jump occurs.

Firstly, we are interested in computing the generator equation of the form

$$Af(x) = \Xi f(x) + \lambda(x) \int_{y \in E} [f(y) - f(x)]Q(x, dy) \tag{5.4.2}$$

which was given in (5.2.16). Remember that $f(x)$ is some value function and $x = (n, \tau)$. The deterministic component of the process is

$$\Xi f(x) = \frac{d}{d\tau} f(n, \tau). \tag{5.4.3}$$

Notice that the hazard rate $\lambda(x)$ is now a function of the state of the process, unlike in a poison process which has a constant hazard rate.

In most applications, particularly in communication systems, the hazard rate is independent of the number of jumps that has occurred so far. Hence, $\lambda(x) = \lambda(\tau)$, and for a gamma density $\Gamma(k, \lambda)$, the hazard rate is $\lambda(\tau) = \lambda^k \tau/(\lambda\tau + 1)$. The gamma distribution is a family of curves based on two parameters (k, λ).

For the second term in (5.4.2), the transition measure $Q(x, dy)$, describing the distribution of the next position x_t following a jump at t from a position x_{t-}, is given as $\delta_{n+1,0}$ for any $x = (n, \tau)$, where δ_z is a Dirac measure at $z \in E$.

Thus,

$$\lambda(x) \int_{y \in E} [f(y) - f(x)] Q(x, dy) = \lambda(\tau)\{f(n+1, 0) - f(n, \tau)\}. \tag{5.4.4}$$

Hence, by combining equations (5.4.3) and (5.4.4), the generator of a renewal process is given as

$$Af(n, \tau) = \frac{d}{d\tau} f(n, \tau) + \lambda(\tau)\{f(n+1, 0) - f(n, \tau)\}. \tag{5.4.5}$$

Now consider the finite time process with a typical state (n, τ, t), where n is the number of renewals that already occurred, τ is the time since the last jump and t is the total elapsed time since we started observing this process. In other words, we observe the process in the interval $[0, t[$ and we are now at time t. We wish to continue observing this process up to a finite time T. Whence $t \in [0, T]$.

Let I be a countable set. Set $X_t = (x_t, t) = (n_t, \tau_t, t)$.

The process $\{X_t\}$ has generator given as $(\partial/\partial t) + A$ where A is given by (5.4.5) with no time dependent coefficients. Using the exit–time hyperbolic equation (??) V is in the domain of the generator if and only if

$$\frac{dV}{dt}(x, t) + AV(x, t) = 0, \quad (n, \tau, t) \in I \times R_+ \times [0, T]. \tag{5.4.6}$$

From the Dynkin formula (Dynkin 1965)

$$E_{x,t}[V(x_T, T)] = V(x, t) + E_{x,t} \int_t^T \left(\frac{dV}{ds} + AV \right)(x_s, s) ds \tag{5.4.7}$$

where $x = (n, \tau)$. Because of (5.4.6) the second term in (5.4.7) vanishes, so the process $\{V(x_t, t)\}$ is a martingle. It has constant expectation. Hence,

$$V(x_t, t) = E_{x,t} V(x_T, T) \tag{5.4.8}$$

but we know that

$$V(x, t) = E_{x,t} \left\{ \int_t^T f(x_\tau, \tau) d\tau + \Omega(x_T) \right\}. \tag{5.4.9}$$

Suppose we set $f = 0, \Omega(x_T) = n$ for our renewal process. Then, according to our condition for the martingale (5.4.8),

$$V(x_t, t) = E_{x,t} [n_T]. \tag{5.4.10}$$

In most communication systems uncertain demand in the services creates a very difficult problem for modeling. Here we are going to model a M/G/1/N queue incorporating holding and service cost. We shall regard the process of servicing a job (or call) as that of making an investment. Here it is assumed that we can alter the speed at which messages are transmitted at each channel of communication system. We assumed that there is just a single channel with a sink that drains messages at the rate u. This provides us with the flexibility of controlling the rate at which messages are transmitted.

With these assumptions it follows that the total investment needed to complete a project is the service time S of each call, whose distribution we know. As shown in figure (5.2) when the investment ζ_t reaches the level S a job is completed and the investment restarts on a new call if the system is non–empty. Let c_t denotes the number service completions in the interval $[0, t)$. Let d_t be the demands in this interval. We assume that when the system is idle there is no penalty.

Figure 5.2: Cumulative investment in a single call

5.4.1 Dynamic of $M/G/1/N$ service process

In most practical applications the service time S has distribution function G that is given by

$$P\{S < s\} = G(s) = \begin{cases} \int_0^s g(r)dr & s \leq D \\ 1 & s > D \end{cases}$$

where $g(.)$ is a bounded continuous function. D is a given

constant, that specifies the maximum expenditure rate. The hazard rate $\mu(s)$

is now given as

$$\mu(s) = \frac{g(s)}{(1 - G(s))}, \quad s \in [0, D], \tag{5.4.11}$$

where S has a probability density g on $(S < D)$.

The hazard rate can be regarded as the mean *completion rate* of the job due to

a constant expenditure of *units of service* at a unit rate of

service. We are assuming here that the units of service is

time-dependent, similar to that of S.

Therefore,

$$\begin{aligned} u(t)\mu(t) \quad = \quad & \text{the mean completion rate} \\ & \text{due to expenditure at rate } u(t). \end{aligned}$$

Suppose that we are interested in varying the completion rate of jobs. Then (5.4.12) helps in the formulation of the dynamic structure of the

service process. Variation in service rate arises mostly in *control problem,* where an optimisation strategy requires one to choose a service rate $u(t) \in [0, 1]$ so as to minimise a certain objective cost.

Since $u(t)$ is the rate of service, we can compute the amount of service that can be *spent* on a job after time τ. Let this be $\zeta(\tau)$. Then

$$\begin{aligned} \zeta(\tau) \quad = \quad & \int_0^\tau u(s)ds \quad \text{mean cumulative} \\ & \text{expenditure at time } \tau \end{aligned}$$

When $\zeta(t)$ increases until it reaches S, the current job is completed and the number of jobs in the system jumps from x_t to $(x_t - 1)$. In between service

completions the cumulative expenditure satisfy the following differential equation

$$\frac{d}{dt}\zeta(t) = u(t). \tag{5.4.12}$$

This means that a call is completed when the cumulative expenditure ζ reaches the level S, at which point ζ is reset to zero.

The effect of the above framework entails that for a fixed service time S, the current job under service is completed quicker if the rate of service $u(t)$ is increased.

5.4.2 Generator of M/G/1/N model

To compute the generator of this system, we need to convert $\{x_t)$ into a Markov process. We create a two–dimensional process $x_t = (n_t, \zeta_t)$, n_t is number of jobs in the system (including the one currently under service) and ζ_t is the expenditure made on the current job since the last service completion.

In addition, for each demand a holding cost h is incurred for each unit of time the job is in the system until its service is completed. There is also an expenditure cost at a rate of r per unit of time that a job is being processed. Let $s(n_t, \zeta_t)$ be the combined expenditure cost. Then

$$s(n_t, u_t) = s(x_t) = h(m_t) + r(u_t), \;\; m > 0, \;\; u \in [0,1]\,. \tag{5.4.13}$$

Let σ be the interest rate that is set on this cost then the value of the capital at time zero of the total expenditure over an infinite horizon is

$$E\int_0^\infty \exp^{-\sigma t} s(x_t)dt. \tag{5.4.14}$$

In some optimisation problem, to be seen later, one objective might be to choose the rate of expenditure in such a way that the holding cost becomes very close to the origin.

Using the idea that has been given earlier the discounting problem can be transformed into a *killed* process with semigroup operator Λ_t. Here, it is assumed that the process is killed at state $x_t \in \{\Delta\}$ after an exponentially distributed random variable with rate σ, where Δ is a cemetary state. The state space of the killed process is now $E_\Delta = E \cup \{\Delta\}$. To get the generator of this process, we let $x_t = (n, \zeta)$ and ask the question," What will become of $x_{t+\Delta t}$ after a small

infinitesimal time

interval Δt

For a bounded measurable function $f : E_\Delta \to R$ we have the expectation

$$\begin{aligned} E_{(n,\zeta)} f(x_{(t+\Delta t)}) &= \lambda \Delta t f(n+1, \zeta u \Delta t) + \mu u \Delta t f(n-1, s) + \{1 - (\lambda \Delta t \quad (5.4.15) \\ &+ \mu u \Delta t + \sigma \Delta t.)\} f(n, \zeta + u\Delta t) + 0(\Delta t), \quad \text{for } s \in [0, \Delta t]. \quad (5.4.16) \end{aligned}$$

Define $A_{\Delta t}$ as

$$A_{\Delta t} = \frac{E_{(n,\zeta)} f(x_{(t+\Delta t)}) - f(n,\zeta)}{\Delta t},$$

so that

$$\begin{aligned} A_{\Delta t} &= \lambda f(n+1, \zeta + u\Delta t) + \mu u f(n-1, s) + \frac{f(n, \zeta + u\Delta t) - f(n,\zeta)}{\Delta t} \\ &+ (\lambda + \mu u + \sigma) f(n, \zeta + u\Delta t) + 0(\Delta t), \quad \text{for } s \in [0, \Delta t]. \end{aligned}$$

Assuming that $f(.)$ is differentiable with respect to ζ then the generator of the process $Af(n,\zeta)$ is given as

$$Af(n,\zeta) = \lim_{(\Delta t \to 0)} A_{\Delta t}.$$

By noticing that

$$\lim_{(\Delta t \to 0)} \frac{f(n, \zeta + u\Delta t) - f(n,\zeta)}{\Delta t} = u \frac{d}{d\zeta} f(n,\zeta),$$

the limit reduces to

$$Af(n,\zeta) = \lambda f(n+1,\zeta) + \mu u f(n-1, 0) + u \frac{d}{d\zeta} f(n,\zeta) - (\lambda + \mu u + \sigma) f(n,\zeta).$$

We can incorporate the cost in (5.4.14). Let $x = (n_0, \zeta_0)$. By noticing that the expenditure at rate u is

$$V(x) = E_x \int_0^\infty s(x_t) I_{(t<\tau)} dt, \quad (5.4.17)$$

where $\tau = \inf\{t : x_t = \Delta.\}$ Δ is the cemetery state and we define $s(\Delta) = 0$. Because of $P_{t\to\infty}\{x_t \to \Delta\} = 1$ we have that $V(x)$ defined by (5.4.17) is the unique solution of the equation

$$Av(x) + s(x) = 0, \quad (5.4.18)$$

with boundary condition

$$v(n, D) = v(n-1, 0). \tag{5.4.19}$$

Let $x_t = (n_t, \tau_t)$, where at time t, n_t is the number of jobs in the systems and τ_t is the time that the server has already been spent on the current job $(1 \leq n_t \leq N, 0 \leq \tau_t \leq D)$. Now consider the finite time process with a typical state $x_t = (n, \tau)$, where n is the number of jobs that are currently in the system, τ is the time that has been spent processing the current job. In addition, consider a modified process with state $X_t = (x_t, t)$. Both processes $\{x_t\}$ and $\{X_t\}$ are Markov.

Take a typical state $(x, t) = (k, \tau, t), t \in (0, T]$ define

$$V(x, t) = E_{(x,t)}\left[\int_t^T f(x_l, l)dl + \phi_{k_T}(\tau_T)\right], \tag{5.4.20}$$

where $\phi(k, \tau, T) = \phi_k(\tau)$ and no discounting, $\alpha = 0$.

In control theory we call $f(x_l, l)$ a *running cost function* and $\phi_{k_T}(\tau_T)$ a *terminal cost function.* If $\phi_{k_T}(\tau_T) = 0$, then the problem is said to be in *Langrange form* and if $s(x_s) = 0$, the problem is in the *Mayer form.* These terms come from calculus of variation (Bardi 1995, Bardi 1997).

Using the Mayer form we can obtain the expected queue length at any time t, $E[q_t]$, by making the following substitutions:

$$\phi_k(.) = k-1, \ k = 1, \ldots, N, \tag{5.4.21}$$

$$\phi_0 = 0, \tag{5.4.22}$$

$$f_k(.) = 0. \ k = 0, \ldots, N, \tag{5.4.23}$$

for $\tau \in [0, D]$.

A complete generator for the queueing can be written as follows.

1. For $k = N$: the buffer is full and all arrivals are rejected. Thus, $\lambda(N, \tau, t) = 0$ and the generator equation (5.4.18) becomes

$$\frac{\partial}{\partial t}V_N(\tau, t) + u\frac{\partial}{\partial \tau}V_N(\tau, t)\mu u + V_{N-1}(0, t) - \mu u V_N(\tau, t) + f_N(\tau, t) = 0.$$

2. For $k = 1, 2, \ldots, N-1$: we have

$$\frac{\partial}{\partial t}V_k(\tau,t) + u\frac{\partial}{\partial \tau}V_k(\tau,t)\lambda V_{k+1}(\tau,t) + \mu u V_{k-1}(0,t) - (\lambda + \mu u)V_k(\tau,t) + f_k(\tau,t) = 0.$$

3. For $k = 0$: there is no service, so that

$$\frac{\partial}{\partial \tau}V_0(\tau,0) = 0,$$

$$\frac{d}{dt}V_0(t) + \lambda\{V_1(0,t) - V_0(t)\} + f_0(t) = 0, \tag{5.4.24}$$

where

$$V_0(t) = V_0(0,t).$$

The boundary conditions are

$$V_k(\tau,T) = \phi_k(\tau)\ k = 1,\ldots,N, \tau \in [0,D], \tag{5.4.25}$$

$$V_0(T) = \phi_0 \tag{5.4.26}$$

$$V_k(D,t) = V_{k-1}(0,t)\ k = 2,\ldots,N,\ \ t \in [0,D], \tag{5.4.27}$$

$$V_1(D,t) = V_0(t). \tag{5.4.28}$$

5.4.3 Generator two–server $M/G/2/N$ queue

Using similar ideas from $M/G/1/N$ queue, now let ζ denote the time elapse since the last departure from the second server. Assume that second server has a general density with hazard rate $\psi(\zeta)$ which is given.

$$\frac{d\zeta}{dt} = w. \tag{5.4.29}$$

Put $X_t = (k, \tau_t, \zeta_t, t)$ assume the boundary conditions

$$V_k(\tau, D_2, t) = V_{k-1}(\tau, 0, t), \tag{5.4.30}$$

$$V_k(D_1, \zeta, t) = V_{k-1}(0, \zeta, t). \tag{5.4.31}$$

The process $\{X_t\}$ is Markov with generators:

1. For $k = N$: the buffer is full and all arrivals are rejected. Thus, $\lambda = 0$ and

the generator equation becomes

$$\frac{\partial}{\partial t}V_N(\tau,\zeta,t)+u\frac{\partial}{\partial \tau}V_N(\tau,\zeta,t)+w\frac{\partial}{\partial \zeta}V_N(\tau,\zeta,t)+\psi(\zeta)wV_{N-1}(\tau,0,t)$$
$$+\mu(\tau)uV_{N-1}(0,\zeta,t)-[\psi(\zeta)w+\mu(\tau)u]\,V_N(\tau,\zeta,t)+f_N(\tau,\zeta,t)=0.$$

2. For $k=3,\ldots,N-1$: we have

$$\frac{\partial}{\partial t}V_k(\tau,t)+u\frac{\partial}{\partial \tau}V_k(\tau,\zeta,t)+w\frac{\partial}{\partial \zeta}V_k(\tau,\zeta,t)+\psi(\zeta)wV_{k-1}(\tau,0,t)+\lambda V_{k+1}(\tau,\zeta,t)$$
$$+\mu uV_{k-1}(0,\zeta,t)-[\lambda(\zeta)w+\mu(\tau)u+\lambda]\,V_k(\tau,\zeta,t)+f_k(\tau,\zeta,t)=0.$$

3. For $k=2$:

$$\frac{\partial}{\partial t}V_2(\tau,\zeta,t)+u\frac{\partial}{\partial \tau}V_2(\tau,\zeta,t)+w\frac{\partial}{\partial \zeta}V_2(\tau,\zeta,t)+\psi(\zeta)wy(\tau,t)+\omega V_3(\tau,\zeta,t)$$
$$+\mu(\tau)uy(\zeta,t)-(\psi(\zeta)w+\mu(\tau)u+\lambda)V_2(\tau,\zeta,t)+f_2(\tau,\zeta,t)=0,$$

where

$$y(\tau,t)=V_1(\tau,0,t).$$

4. For $k=1$: **only one server is working (i.e.** $u\neq 0, w=0$:)

$$\frac{\partial}{\partial t}y(\tau,t)+u\frac{\partial}{\partial \tau}y(\tau,t)+\lambda V_2(\tau,0,t)+\mu(\tau)uz(t)-[\mu(\tau)u+\lambda]\,y(\tau,t)+\bar{f}_1(\tau,t)=0.$$

where

$$z(t)=V_0(0,0,t).$$

5. For $k=0$: there is no service, so that

$$u=w=0.$$

$$\frac{\partial}{\partial t}z(t)++\lambda\{y(0,t)-z(t)\}=0, \qquad (5.4.32)$$

with boundary conditions

$$V_k(\tau,\zeta,T) = \phi_k(\tau,\zeta), \tag{5.4.33}$$

$$k=1,\ldots,N,\tau\in[0,D_2],\zeta\in[0,D_2], \tag{5.4.34}$$

$$V_0(T) = \phi_0 \tag{5.4.35}$$

$$V_k(\tau,D_2,t) = V_{k-1}(\tau,0,t), \tag{5.4.36}$$

$$V_k(D_1,\zeta,t) = V_{k-1}(0,\zeta,t), \tag{5.4.37}$$

$$k=3,\ldots,N,\ \tau\in[0,D_1],\zeta\in[0,D_2]., \tag{5.4.38}$$

$$V_2(D_1,\zeta,t)=y(\zeta,t), \qquad V_2(\tau,D_2,t)=y(\tau,t), \tag{5.4.39}$$

$$\tau\in[0,D_1],\zeta\in[0,D_2]., \tag{5.4.40}$$

$$y(D_1,t) = z(t), \tag{5.4.41}$$

$$y(\tau,T) = \bar{\phi}(\tau), \tag{5.4.42}$$

$$z(T) = \bar{\bar{\phi}}. \tag{5.4.43}$$

5.4.4 Generator of $G/G/1/N$ queue

Using similar ideas from $M/G/1/N$ queue, let ζ denote the time elapse since the last arrival. Assume that arrivals have a general density with hazard rate $\lambda(\zeta)$ which is given. Let

$$\frac{d\zeta}{dt} = w. \tag{5.4.44}$$

Put $X_t=(k,\tau_t,\zeta_t,t)$ assume the boundary conditions

$$V_k(\tau,D_2,t) = V_{k+1}(\tau,0,t) \tag{5.4.45}$$

$$V_k(D_1,\zeta,t) = V_{k-1}(0,\zeta,t). \tag{5.4.46}$$

The process $\{X_t\}$ is Markov with generators:

1. For $k=N$: the buffer is full and all arrivals are rejected. Thus, $\lambda=0$ and the generator equation becomes

$$\frac{\partial}{\partial t}V_N(\tau,0,t)+u\frac{\partial}{\partial\tau}V_N(\tau,0,t)\mu uV_{N-1}(0,0,t)-\mu uV_N(\tau,0,t)=0.$$

2. For $k=1,2,\ldots,N-1$: we have

$$\frac{\partial}{\partial t}V_k(\tau,t)+u\frac{\partial}{\partial\tau}V_k(\tau,\zeta,t)+w\frac{\partial}{\partial\zeta}V_k(\tau,\zeta,t)+\lambda(\zeta)wV_{k+1}(\tau,0,t)$$
$$+\mu uV_{k-1}(0,t)-(\lambda(\zeta)w+\mu(\tau)u)V_k(\tau,t)=0.$$

3. For $k = 0$: there is no service, so that

$$\frac{\partial}{\partial \tau} V_0(\tau, ., .) = 0,$$

$$\frac{\partial}{\partial t} V_0(0, \zeta, t) + w \frac{\partial}{\partial \zeta} V_0(0, \zeta, t) + \lambda(\zeta) w \{V_1(0, \zeta, t) - V_0(0, \zeta, t)\} = 0, \quad (5.4.47)$$

with boundary conditions

$$\begin{aligned}
V_k(\tau, \zeta, T) &= \phi_k(\tau, \zeta), && (5.4.48)\\
&\quad k = 1, \ldots, N, \tau \in [0, D_2], \zeta \in [0, D_2], && (5.4.49)\\
V_0(T) &= \phi_0 && (5.4.50)\\
V_k(\tau, D_2, t) &= V_{k+1}(\tau, 0, t), && (5.4.51)\\
V_k(D_1, \zeta, t) &= V_{k-1}(0, \zeta, t), && (5.4.52)\\
&\quad k = 2, \ldots, N, \ \tau \in [0, D_1], \zeta \in [0, D_2]. && (5.4.53)\\
& && (5.4.54)
\end{aligned}$$

5.5 Summary

In this chapter we have presented the renewal properties of PDPs, including the connection of their associated generators. It turns out that there is a direct relationship between expected values of certain functionals defined on the PDPs and the generators of certain renewal processes.

In addition, there is a direct link with martingale calculus through the interplay of the semigroups, generators, renewal functions. This development is entirely new and it will have, we hope, considerable implications for future research in PDP/queue.

We have applied PDP theory to single server queueing systems $M/G/1/N, G/G/1/N$ and two–servers case. Also we have extended this formulation to those of network queues. In each case, the performance parameters of interest are represented as some cost functionals of the PDP. These functionals appear in the generator of the PDP. Depending on the application and the performance measures of interest we can tune the generator and set the boundary accordingly. We have shown how this is done to obtain the blocking probabilities.

We have also presented a truncation method of converting the general service

distribution G to a distribution on a finite domain. We have shown how, using the link between the generator of a renewal process and the renewal function, one can set the boundary conditions. The procedure for setting the boundary conditions will be used in chapter (6) to develop methods for solving the transient queueing problem.

The criticism that has been made of PDP that too much attention is given to broad questions about existence of optimality condition and so little on computational issues are now being addressed. With the present set up fairly good computation models are emerging, bringing the subject to the domain of computer science, with good consequences for communication systems.

Chapter 6

Computation and simulation of $M/G/1/N$ queues

6.1 Introduction

In the previous chapter we used the PDP Markov theory to develop a new generator of a $M/G/1/N$ queue. This chapter is concerned with computational and simulation issues arising from $M/G/1/N$. This chapter treats the new transient model, compares its results with steady state analytic results from mean value analysis of $M/G/1/N$. We have also developed several types of methods for obtaining steady state results from simulation outputs. We have also examined results from some simulation packages, as well as developing new ones.

It is always difficult to describe with adequate conciseness and clarity the scientific status of simulation of complex systems like computer systems and queues. But if one must define the status of this simulation, the answer may be this: the creation of statistical environment that emulate the response of a queueing system(to some input variables) that is too complex to be understood analytically. IBM has several queueing simulation packages which were used.

To deduce the steady state solution from transient solution one will have to solve the new partial differential equations and obtain the solution for very large time step(T). As we shall see, the solution converges to a stationary value, which we can compare with known analytical and simulation steady results.

The computational methods developed help us to understand the response of the queue length with respect to the system parameters and boundary conditions. It

widely understood that the relative improvement in the understanding of the system behaviour is not generally achieved until the system best capabilities are determined and tradeoffs between various changes in system conditions fully understood. Bearing this in mind, we were very open–minded in our investigation, regarding the numerical study as a serious enterprise that can lend itself to a novel contribution to PDP study on its own right.

As the variation of parameter sets increases, such tradeoffs are likely to become complex and less easily quantified. Hence, there is more reliance on the intuition of the researcher and he or her to express preference throughout the testing of the new system. This is exactly what is happening here. My intuition is that by restricting ourselves to a class of Erlang systems we shall be able to appreciate the contribution made and relax the rigidity of the mathematical problem posed.

There are three different methods used:

1. Our solution of $M/G/1/N$ queue based on the generations we developed in chapter (5).

2. Our solution of $M/G/1/N$ using output simulation of the queue.

3. The analytic results which are found in literature of queueing systems.

In the next few sections we show one methods of solving the partial differential equations resulting from the Markov generator of $M/G/1/N$. This is an implicit algorithm which is developed from on a finite difference scheme. A second method, which is more application to the modelling data communication of chapter (8), is developed using an *upwind scheme,* a very popular method[1]. We have adopted one distribution for G, an Erlang distribution, which we denote as $E_k(\mu t)$, where k is a special parameter that regulates the shape of this distribution. An important property of an Erlang distribution is that it can be used to approximate nearly any distribution of practical interest.

6.2 Numerical solution of $M/G/1/N$ queue

From the generator equations in chapter (5), we make the transformation $t = T - t$. Then, we have

[1]This is method is associated with Peter Lax, a famous mathematician in Courant Institute of Mathematical Sciences, New York.

1. For $k = N$:

$$\left.\begin{array}{lcl} \frac{\partial}{\partial t}V_N(\tau,t) & - & u\frac{\partial}{\partial \tau}V_N(\tau,t) \\ +\gamma_N(\tau)V_{N-1}(0,t) & + & \alpha_k(\tau)V_N(\tau,t) = f_N(\tau,t). \end{array}\right\} \tag{6.2.1}$$

2. For $k = 2,\ldots,N-1$:

$$\left.\begin{array}{lcl} \frac{\partial}{\partial t}V_k(\tau,t) & - & u\frac{\partial}{\partial \tau}V_k(\tau,t) + \beta_k V_{k+1}(\tau,t) + \\ \gamma_k(\tau)V_{k-1}(0,t) & + & \alpha_k(\tau)V_k(\tau,t) = f_k(\tau,t). \end{array}\right\} \tag{6.2.2}$$

3. For $k = 1$:

$$\left.\begin{array}{lcl} \frac{\partial}{\partial t}V_1(\tau,t) & - & u\frac{\partial}{\partial \tau}V_1(\tau,t) \\ & + & \beta_1 V_2(\tau,t) + \gamma_2(\tau)W(t) \\ & + & \alpha_1(\tau)V_1(\tau,t) = f_1(\tau,t). \end{array}\right\} \tag{6.2.3}$$

4. For $k = 0$:

$$\frac{d}{dt}w(t) - \lambda V_1(0,t) + \lambda w(t) = f_0(t), \tag{6.2.4}$$

where

$$\alpha_k(\tau) = \lambda + u\mu_k(\tau), \quad \beta_k = -\lambda, \quad \gamma_k(\tau) = -u\mu_k(\tau), \tag{6.2.5}$$

$$\beta_N = 0, \text{for} \quad t > 0, \quad k = 1,\ldots,N, \tau \in [0,D], \tag{6.2.6}$$

$$w(0) = 0, \tag{6.2.7}$$

$$V_k(D,t) = V_{k-1}(0,t) \;\; k = 2,\ldots,N, \;\; t \in [0,D], \tag{6.2.8}$$

$$V_1(D,t) = w(t), \tag{6.2.9}$$

$$\text{for} = t \geq 0. \tag{6.2.10}$$

This is a system of N+1 hyperbolic differential equations, from which we can obtain, for each k, a characteristic line in the (τ,t)–plane given by $\tau = \tau_t$, where $\tau(t)$ is the solution of the differential equation

$$\frac{d\tau}{dt} = -u; \longrightarrow \tau + ut = c. \tag{6.2.11}$$

$$\frac{d}{dt}v_k(c-ut,t) = \left(\frac{\partial v_k}{\partial t} - u\frac{\partial v_k}{\partial \tau}\right)\}_{x=c-ut}. \tag{6.2.12}$$

Define the mesh (τ_i, t_j):

$$\tau_i = (i-1)uh, \;\; i = 1,2,\ldots,M,$$

where

$$h = \frac{D}{(M-1)u},$$

$$t_j = (j-1)h, \quad j = 1, 2, \ldots, M.$$

Define $w_j \approx w(t_j)$ and $v_{k,i,j} \approx v_k(\tau_i, t_j)$. By using the Trapezoidal rule (Iserles 1986, Zachmanoglu and Thoe 1986), one can obtain

1.

$$\begin{aligned} \frac{V_{N,i,j+1} - V_{N,i+1,j}}{h} &+ \frac{\alpha_{N,i}}{2}(V_{N,i,j+1} + V_{N,i+1,j}) \\ +\frac{\gamma_{N,i}}{2}(V_{N-1,1,j+1} + V_{N-1,1,j}) &= \frac{1}{2}((f_{N,i,j+1} + f_{N,i+1,j}). \end{aligned}$$

2.

$$\begin{aligned} \frac{V_{k,i,j+1} - V_{k,i+1,j}}{h} &+ \frac{\alpha_{k,i}}{2}(V_{k,i,j+1} + V_{k,i+1,j}) + \frac{\beta_k h}{2}(V_{k+1,i,j+1} + V_{k+1,i+1,j}) \\ +\frac{\gamma_{k,i}}{2}(V_{k-1,1,j+1} + V_{k-1,1,j}) &= \frac{1}{2}((f_{k,i,j+1} + f_{k,i+1,j}). \end{aligned}$$

3.

$$\begin{aligned} \frac{V_{1,i,j+1} - V_{1,i+1,j}}{h} &+ \frac{\alpha_{,i}}{2}(V_{1,i,j+1} + V_{1,i+1,j}) + \frac{\beta_1}{2}(V_{2,i,j+1} + V_{2,i+1,j}) \\ +\frac{\gamma_{1,i}}{2}(w_{j+1} + w_j) &= \frac{1}{2}(f_{1,i,j+1} + f_{1,i+1,j}). \end{aligned}$$

4.

$$\frac{w_{j+1} - w_j}{h} - \frac{\lambda}{2}(V_{1,i,j+1} + V_{1,i+1,j}) + \frac{\lambda}{2}(w_{j+1} + w_j) = \frac{1}{2}(f_{0,j+1} + f_{0,j}).$$

Given

$$\begin{cases} w_1 &= \phi_0, \\ V_{k,i,1} &= \phi_{k,i} \;\; k = 1, 2, \ldots, N, i = 1, 2, \ldots, M, \\ V_{1,M,j+1} &= w_{j+1}, \\ V_{k,M,j+1} &= V_{k-1,1,j+1}, \;\; k = 2, 3, \ldots, N, j = 1, 2, \ldots, . \end{cases}$$

The numerical procedure is as follows:

Given

$$w_j, V_{k,i,j}, k = 1, 2, \ldots, N, i = 1, 2, \ldots, M, j \geq 1,$$

find

$$w_{j+1}, V_{k,i,j}.$$

Step 1: Find w_{k+1} and $V_{k,1,j+1}, k = 1, 2, \ldots, N$,

$$_{j+1} = \frac{(1-(1/2)\lambda h)}{(1+(1/2)\lambda h)} w_j + \frac{(h/2)}{(1+(1/2)\lambda h)} \left(V_{1,1,j} + V_{1,1,j+1}\right),$$

which leads to

$$A\bar{V} = \bar{G}, \tag{6.2.13}$$

where A is an $N \times N$ tridiagonal matrix, $\bar{V}$ and $\bar{G}$ are vectors

$$\bar{V} = [V_{1,1,j+1}, V_{2,1,j+1}, \ldots, V_{N,1,j+1}]^T, \bar{G} = [g_1, g_2, \ldots, g_N]^T$$

with entries that are well defined. Step 2: Find
$V_{k,i,j+1}, k = 1, 2, \ldots, N, i = 1, 2, \ldots, M-1$.

1.

$$\begin{aligned} V_{N,i,j+1} &= \frac{1}{(1+\frac{\alpha_{N,i}h}{2})} \{ (1-\frac{\alpha_{N,i}h}{2}) V_{N,i+1,j} \\ &+ \frac{\gamma_{N,i}}{2}(V_{N-1,1,j+1} + V_{N-1,1,j}) \\ &+ \frac{1}{2}((f_{N,i,j+1} + f_{N,i+1,j}) \} . \end{aligned}$$

2.

$$\begin{aligned} V_{k,i,j+1} &= \frac{1}{(1+\frac{\alpha_{k,1}h}{2})} \{ (1-\frac{\alpha_{k,i}h}{2}) V_{k,i+1,j} \\ &- \frac{\alpha_{k,i}}{2}(V_{k,i,j+1} + V_{k,i+1,j}) + \frac{\beta_k}{2}(V_{k+1,i,j+1} + V_{k+1,i+1,j}) \\ &- \frac{\gamma_{k,i}}{2}(V_{k-1,1,j+1} + V_{k-1,1,j}) \\ &+ \frac{1}{2}((f_{k,i,j+1} + f_{k,i+1,j}) \} . \\ k &= N-1, N-2, \ldots, 2. \end{aligned}$$

3.

$$\begin{aligned} V_{1,i,j+1} &= \frac{1}{(1+\frac{\alpha_{1,1}h}{2})} \{ (1-\frac{\alpha_{1,i}h}{2}) V_{1,i+1,j} \\ &+ \frac{\alpha_{1,i}}{2}(V_{1,i,j+1} + V_{1,i+1,j}) + \frac{\beta_1}{2}(V_{2,i,j+1} + V_{2,i+1,j}) \\ &+ \frac{\gamma_{1,i}}{2}(w_{j+1} + w_j) + \frac{1}{2}(f_{1,i,j+1} + f_{1,i+1,j}) \} . \end{aligned}$$

Step 3: Find $V_{k,M,j+1}$.

$$V_{k,M,j+1} = V_{k-1,1,j+1} \; k = 2, \ldots, N, \tag{6.2.14}$$

$$V_{1,M,j+1} = w_{j+1}. \tag{6.2.15}$$

6.2.1 Modelling service distribution

The Erlang distribution is a special type of gamma distribution, named after A. K. Erlang, who pioneered queueing theory and its application to telephone congestion. The idea is that the service facility consists of a number of stages through which a customer must passed before a job is completed. The next customer can not start service until the previous customer has completed all the stages. The time taken for each stage has an exponential distribution, and the mean time of each stage is the same.

The Erlang distribution, similar to distributions like the Exponential, Normal, Uniform, Weibull, and Chi–square is a distribution representing a continuous random variable. The distribution is named after Agner Krarup Erlang, a Danish mathematician who worked for the Copenhagen Telephone Company on problems including loss and waiting times in telephone calls. A.K. Erlang's first work was published in 1909 and describes the problems in detail.

Erlang distributions are excellent distributions for applications, for fitting the random process of lifetimes. For example, tool life, time to perish (for perishable items) and reliability periods can all be modeled as random variables with low variability and hence use Erlang distributions. What makes Erlang distributions a good choice for lifetime models, as we shall see, is that those Erlang random variables have an increasing failure rate, which means, the longer they have lived, the sooner they would die. Distributions like the exponential and normal do not have this property.

By setting $x = \mu t$, the service time distribution follows an Erlang(k), denoted as E_{k-1}[2] The service time X is distributed as Erlang (k) implies that

$$P[X > t] = E_{k-1}(k\mu t), \;\; t \geq 0. \tag{6.2.16}$$

[2]It is written like that because Erlang (1) is an exponential distribution = E_0 and also note that in our formulation each stage of an k Erlang phases is exponentially distributed with parameter $k\mu$.

Thus, this has probability distribution $F(t)$, given as

$$F(t) = P[X \leq t] = 1 - E_{k-1}(k\mu t) = 1 - \sum_{n=0}^{k-1} (k\mu t)^n e^{-k\mu}/n!, \tag{6.2.17}$$

with density

$$f(t) = k\mu(k\mu t)^{k-1} \left\{ e^{-k\mu t}/(k-1)! \right\}. \tag{6.2.18}$$

It is important to note that although the mean holding time in each stage in our fictitious simulation is $k\mu$ the average holding time in the whole system is

$$\frac{1}{\mu}$$

and the squared coefficient of variation C_X is

$$C_X^2 = \frac{1}{k}$$

which is computed from

$$C_X = \frac{\sigma(X)}{E[X]},$$

where $\sigma(X)$ is the standard deviation of X. Therefore, from interest of renewal theory, the *harzard rate function* $\Lambda(t)$ of X is

$$\Lambda(t) = \frac{f(t)}{1 - F(t)}, \tag{6.2.19}$$

for those values of T for which $F(T) < 1$. This has a useful probabilistic interpretation. Thinking of X as the elapsed service time, it follows from the relation

$$P[X \in (t, t + \Delta t] \,|X > t] \approx \Lambda(t)\Delta t \text{ for small } \Delta t \tag{6.2.20}$$

that $\Lambda(t)\Delta t$ gives approximately the probability that job which has been in service for up to time t will finish service in the next time Δt.

6.3 Simulation of service systems

We generate the following:

1. Generate n arrive sequences with inter–arrival time $1/\lambda$.

2. Generate n service sequences with holding time of mean θ such that the

distribution is $Erlang(k)$, equivalent to the sum of k exponential random variables eah with mean θ/k.

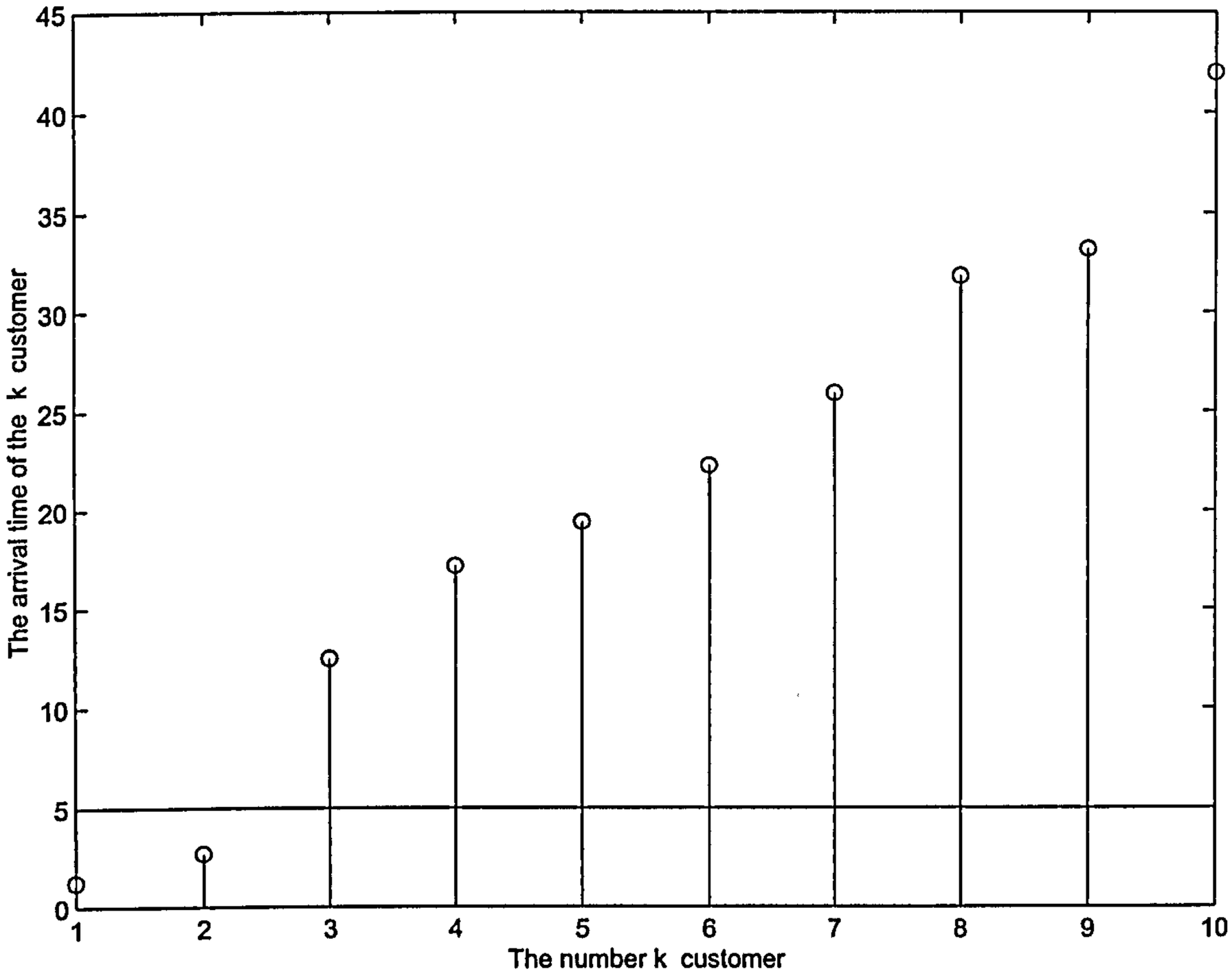


Figure 6.1: Inter arrival times of customers

One particular instance of the arrival jobs are shown in figure (6.1) with the results of the first 10 customers. For our readers using Matlab package we warn that code:

- y1= gamrnd(20, 8/20, 1, 1000);
- y2=gamrnd(1, 8, 1, 1000);

will produce $y1$ as an Erlang–10 distribution with holding time of mean 8 minutes (or any other units) and $y2$ as an exponential distribution with mean holding time 8 minutes.

The practical realisation of y1 is as if we have 20 phases of exponential distributed stages each with mean holding times 8/10, so that the overal mean holding time is 8. Hence, in general each exponential phase has rate $\mu = k/\mu_0$ and the overal system has standard deviation $\mu_0/\sqrt{k}$, $\mu_0 = 8$ the overal holding time in minutes. The plots in figures (6.2, 6.3, 6.4) have the same mean but with $k = 1, k = 2$ and

$k = 5$ respectively. On the same scale in figure (6.5), the plot compares Erlang (20) with normal of mean 8. See how close the two graphs are, showing that for large k, Erlang approximates a normal distribution.

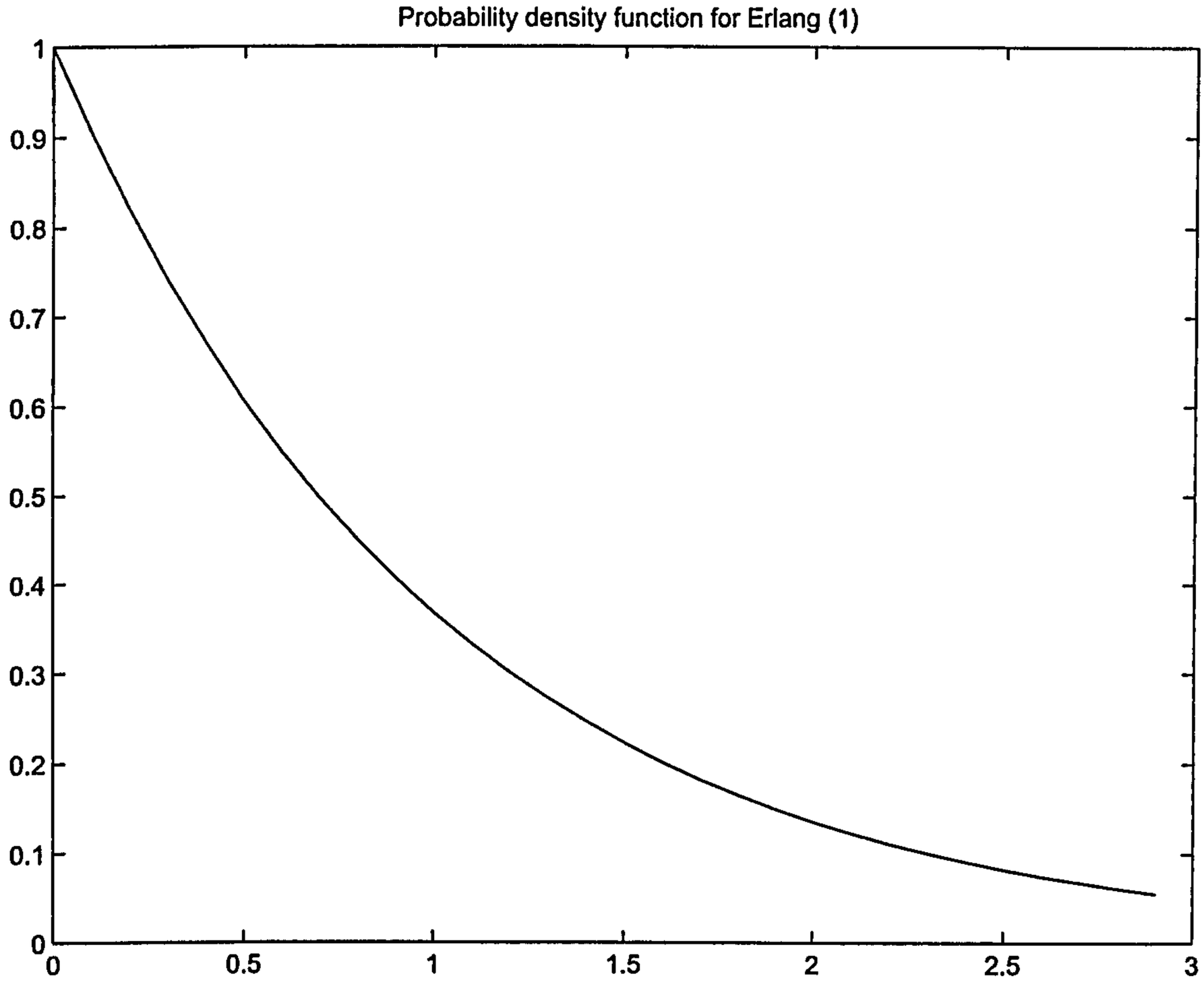


Figure 6.2: Exponential distribtion

6.3.1 Computing the truncation point D

To fix the truncation D in the $M/G/1/N$ computational model, we need to simulate the hazard rate with these parameters and also look at the graphs above. We went through the following procedure:

1. We generate time slots in minutes from $t = 0, 0.1, 0.2, \ldots, 30$. These are 300 time slots.
2. Compute the hazard rate $\Lambda(t)$ with respect to these times.
3. Compute numerically the integral

$$I(t) = \int_0^t \Lambda(s)ds \tag{6.3.1}$$

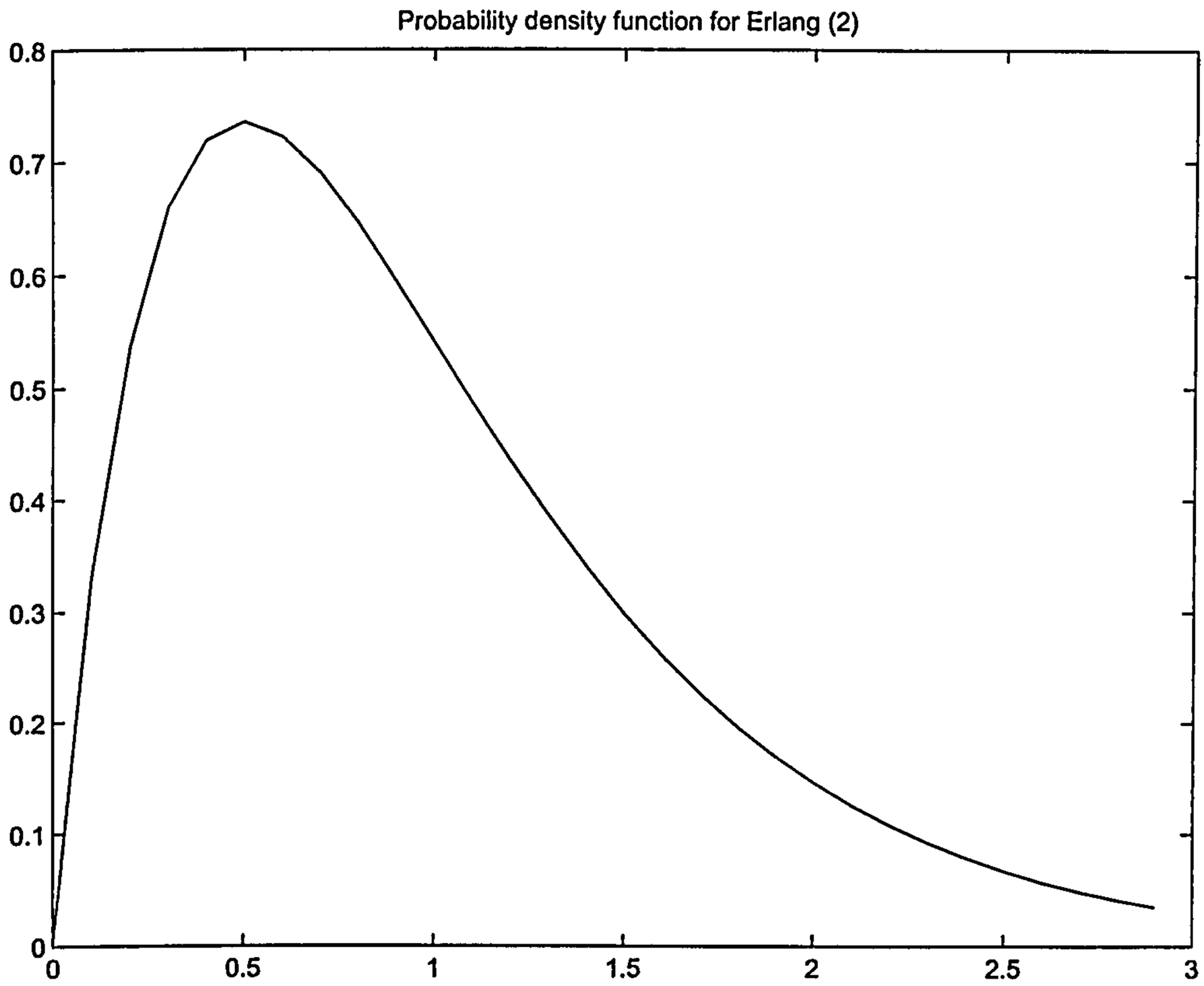


Figure 6.3: Erlang (2) distribtion

for each t. The integral is then computed using Simpson method or Trapezoidal rule (Iserles 1996). Then using renewal theory we compute the survival probability function

$$P(t) = P(X > t, t \geq 0) = \exp\{-I(t)\}. \tag{6.3.2}$$

4. Figure (6.7) shows the plot of the survival function $P(t)$ and from this we can pick the truncation time at $D = 12$, where $P(12) = 2.5049e - 017$, a very negligible failure. Therefore, the assumption that a job that has been in service for up to time $t = 12$ minutes must be completed is reasonable from a probabilistic point of view.

Figure (6.8) shows the result for offer load $\rho = 0.5$, $\mu_0 = 4$ where $q = (0.1, 5, 1, 12, 20, 0, 70, 70)$. As shown the stationary queue length is 0.23. We now maintain $\mu_0 = 4$ and change k to 2. The result of the computation is shown in figure (6.9), where we see that as the queue length averages at 0.3234.

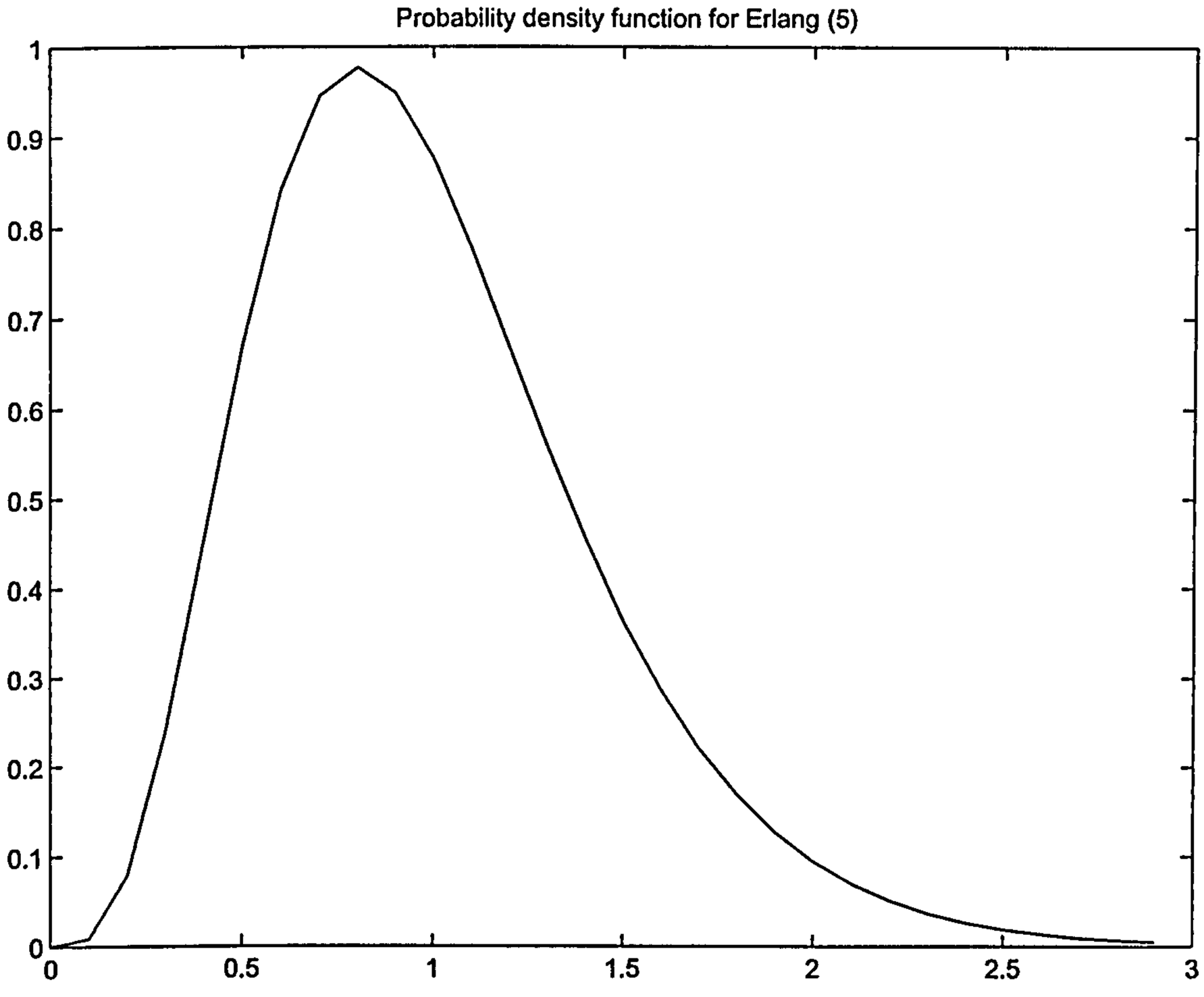


Figure 6.4: Erlang(5) distribution

Figure (6.12) shows that plot of queue length of an $M/G/1/\infty$ queue against different Erlang (k) . As shown the queue length is highest at $k = 1$ which is the exponential distribution and smallest at the highest value k. As k increases the queue length decrease and steadily converges to the queue length of $M/D/1/\infty$, which has a deterministic service time.

6.4 Computing hazard rates

For an Erlang the hazard rate $\Lambda(t)$ is given as

$$\Lambda(t) = \left(\mu^k \frac{t^{k-1}e^{-\mu t}}{(k-1)!}\right) \div \left(\sum_{j=0}^{k-1} \frac{(\mu t)^j}{j!} e^{-\mu t}\right), \tag{6.4.1}$$

where μ is the exponential rate. With respect to defined functions e_n and E_n, this is given as $\mu^k e_{k-1}(\mu t)/E_{k-1}(\mu t)$. In figure (6.11) we illustrate the facts about the hazard rate $\Lambda(t)$ by drawing the hazard rate with respect to time for $k = 1, 5, 10$; note the we have changed notation of $\Lambda(t)$ to $\mu(t)$. In this computation, we use

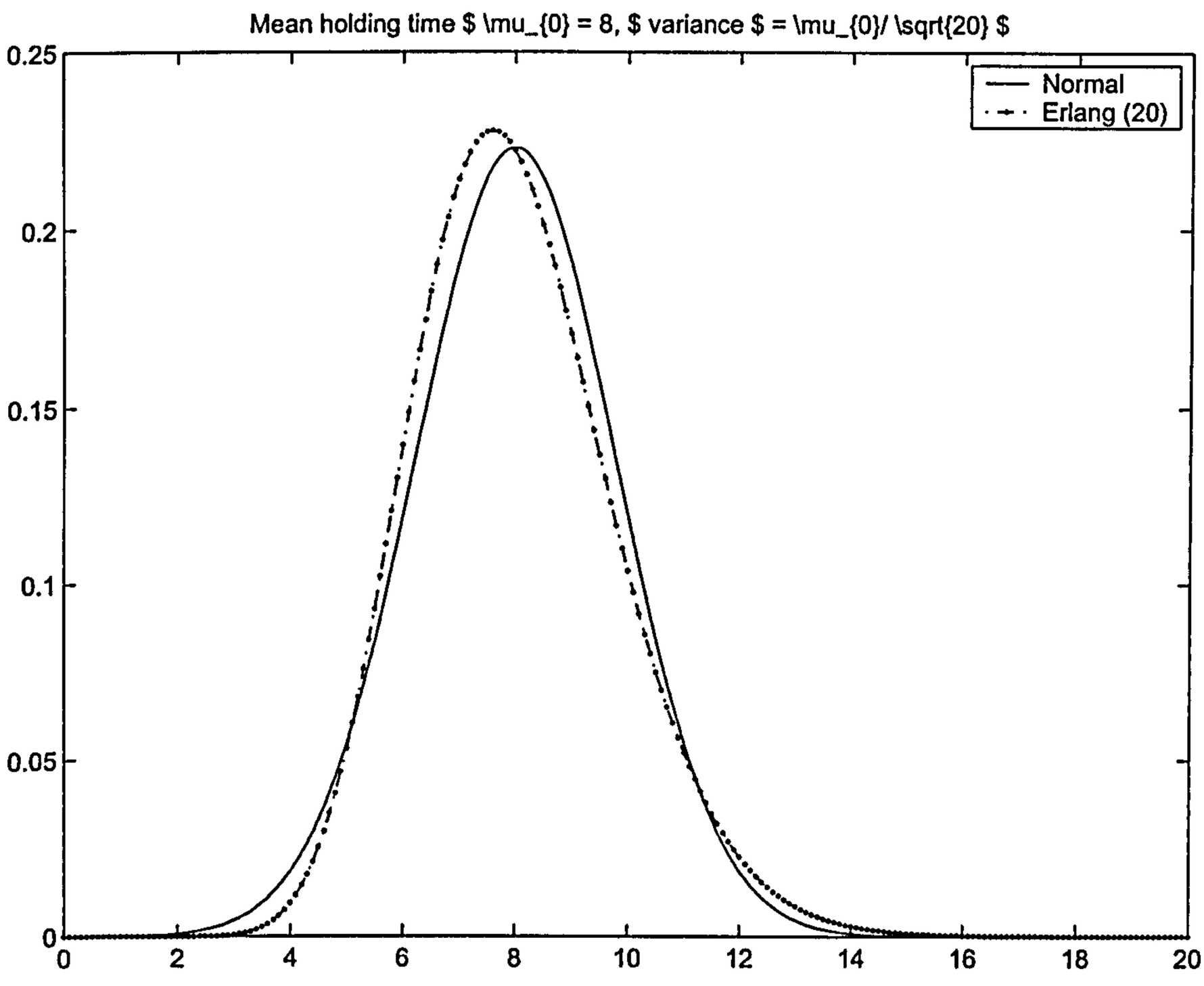


Figure 6.5: Comparing Erlang (20) with normal

keep the mean holding time at each stage of the k exponential phases at $8/k$ so that the overall holding time is 8 minutes.

As expected, the hazard rate is constant for $k = 1$ which corresponds to an exponential distribution. As k increases the hazard rate increases for any fixed time t. This is because, as we mentioned, increases in k reduce the variability below unity, a property that produces such an increase.

6.4.1 Approximation of $M/D/1/N$ model of Davis

There are two ways to test if our results of $M/G/1/N$ are similar to the results obtained in (Davis and Howl 1997):

1. The first method is to set the hazard rate of G equal to zero and set the bound D on the service time, which appears in the distribution of G, to be equal to the deterministic service time D in (Davis and Howl 1987)

2. The second method is to use Erlang distribution with an overall mean

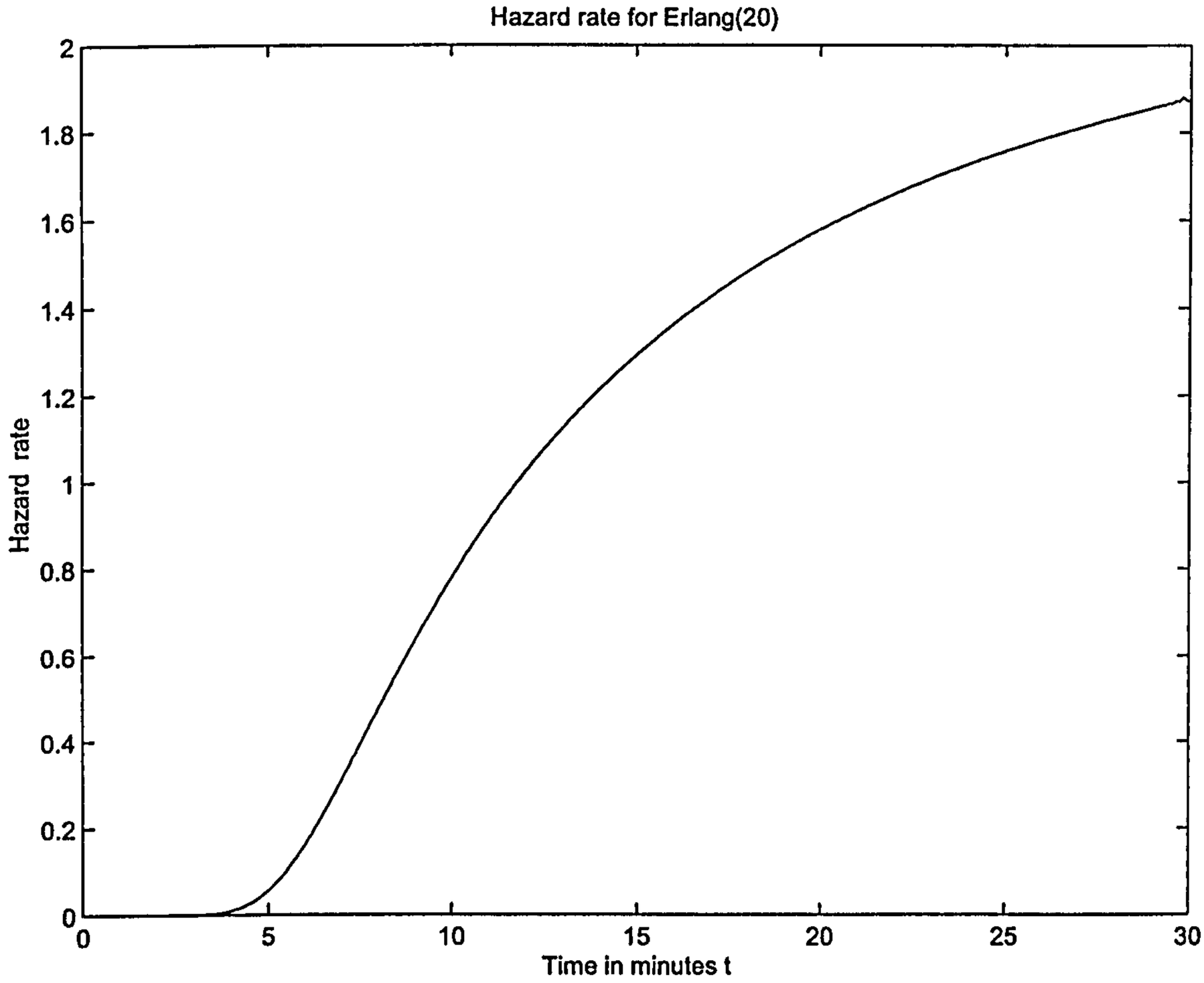


Figure 6.6: Hazard rate of Erlang(20)

holding time equal to D, the service time in Davis's paper. Then the limit of the resulting holding time as $k \to \infty$ is normally distributed with variance $1/k\mu$ that converges to zero. Thus, in principle, the whole systems for large k should approximate $M/D/1/N$ model in Davis's paper.

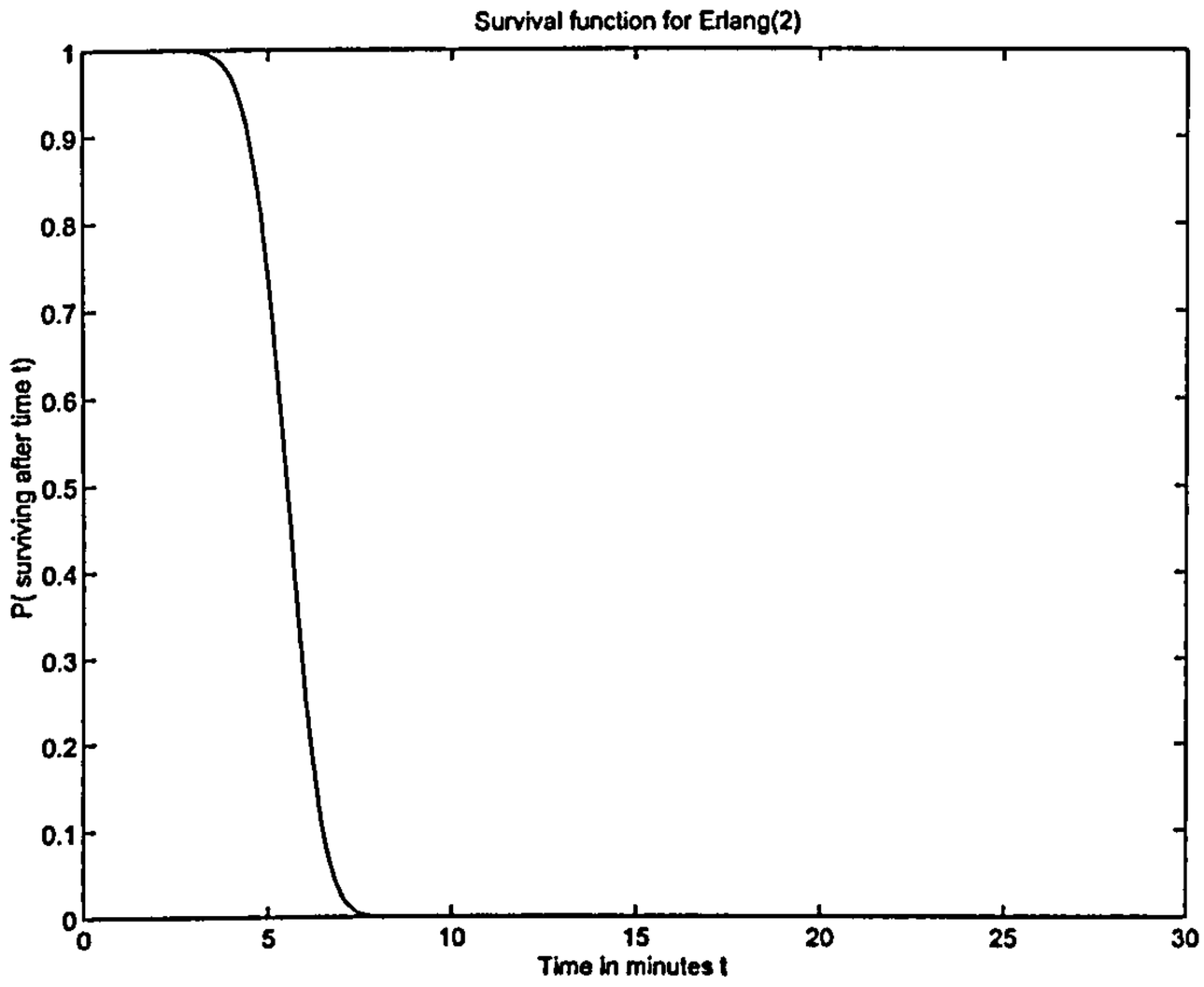


Figure 6.7: Hazard rate of Erlang(20)

By solving the generator equation with the Hazard rate set to zero as explained we have the following results.

λ	D	N	T	Expected queue length after time T
0.1	5	20	50	0.1831
0.1	6	20	50	0.2858
0.1	7	20	50	0.4102
0.1	8	20	50	0.5424
0.1	9	20	50	0.6861
0.1	2	20	50	0.0224
0.1	15	20	50	1.4090
0.05	30	6	100	1.3634
0.05	30	6	300	1.5918
0.05	30	6	400	1.7103

6.5 Simulation software

The Arena modeling system from Systems Modeling Corporation is a flexible and powerful tool that allows analysts to creates animated simulation models that represent virtually any system. The package was first released in 1993 and it employs an object-oriented design for entirely graphical model development. It is similar to SIMULINK graphical interface in MATLAB (Mathworks 1995). Any

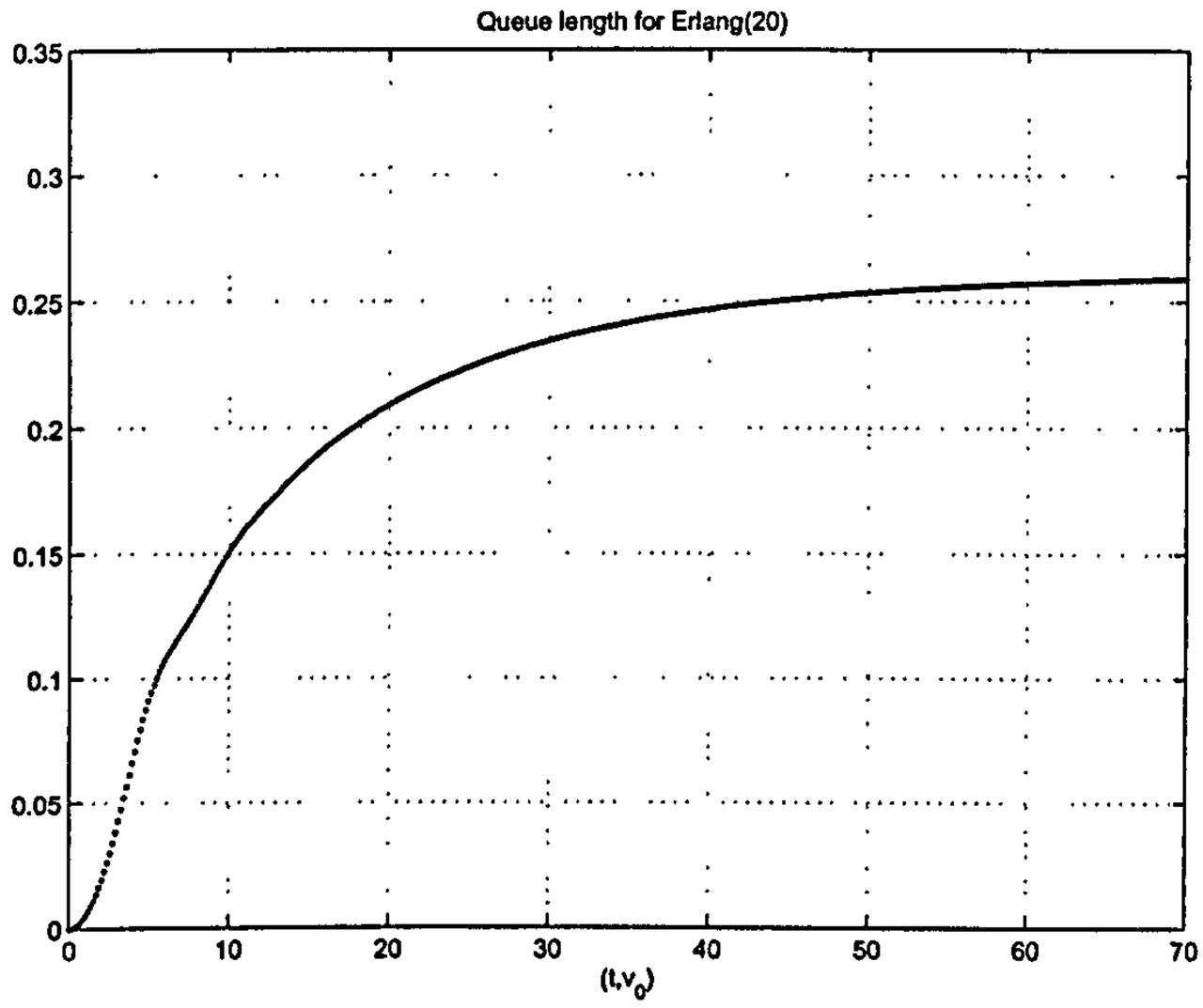


Figure 6.8: Hazard rate of Erlang(20)

analyst using Arena places a graphical objects known as on a layout in order to define system components such as machines, operators, and material handling devices.

SIMAN(SIMulation ANalysis) was developed by Dennis Pegden (Pegden, Sadowski and Shannon 1990) and is distributed by Systems Modelling Corporation[3]. SIMAN got rapid acceptance because it was the first leading simulation language to be available on personal computers and also because its special features for modeling industrial processes in the field of manufacturing, work stations, queues and inventories.

Arena is built on the SIMAN simulation language. After creating a simulation model graphically, Arena automatically generates the underlying SIMAN model used to perform simulation runs. The graphical modules used by simulation analysts to create models are provided automatically with Arena.

These modules can also be designed by the end user to produce a modeling environment that is tailored to a specific application area. Resulting system is a collection of user–created modules that is contained inside a Solution Template that can be shared by other user codes.

The Arena template is the core collection of multiple modules that are provided as part of a single system. This system was designed to provide a general–purpose

[3]This is a company in Pennsylvania, USA.

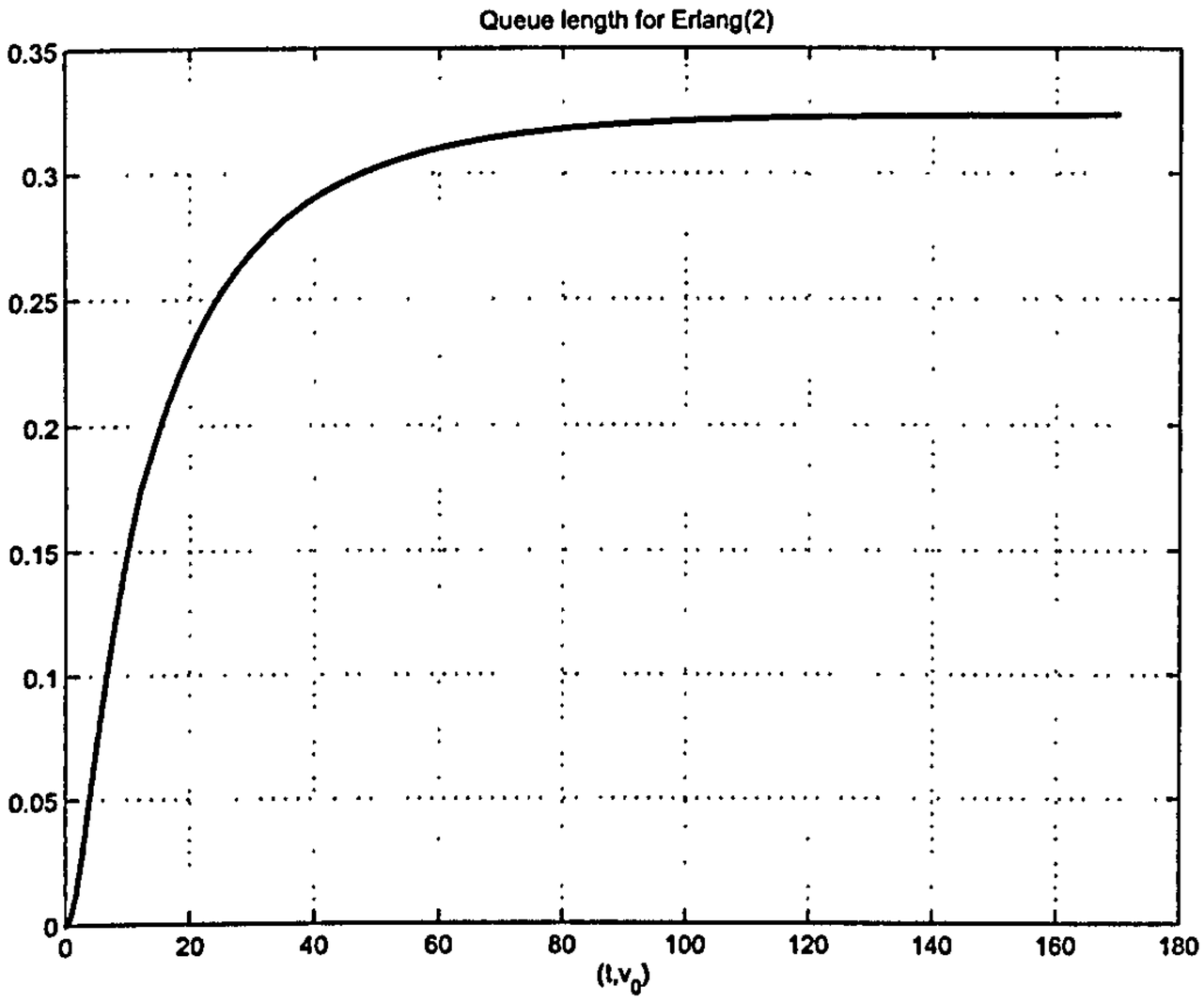


Figure 6.9: Hazard rate of Erlang(20)

collection of modeling features for all types of applications. In addition to providing core features for resources, queueing, inspection, system logic, and external file interface, the Arena system provides modules that are specifically focused on specific aspects of manufacturing and material handling.

Manufacturing–specific modules incorporate such features as machine downtime and maintenance schedules. Material handling applications, modules exist for representing conveyors and various types of transportation devices.

There are three panels that compose the Arena template: the Common panel with modules representing the key simulation processes such as arrivals, service, and departures; the Support panel, containing supplemental modules for specific actions and decision logic; and the Transfer panel, whose modules are used to model the transfer (or flow) of entities through the system.

6.6 Simulation results of $M/G/1/N$ with Arena

Figure (6.12) shows the results of queue length of M/G/1/N queue with various Erlang functions. The results matches well with both theoretical estimated and those obtained From Davis' theory. Arena template, we did the following three basic Arena functions:

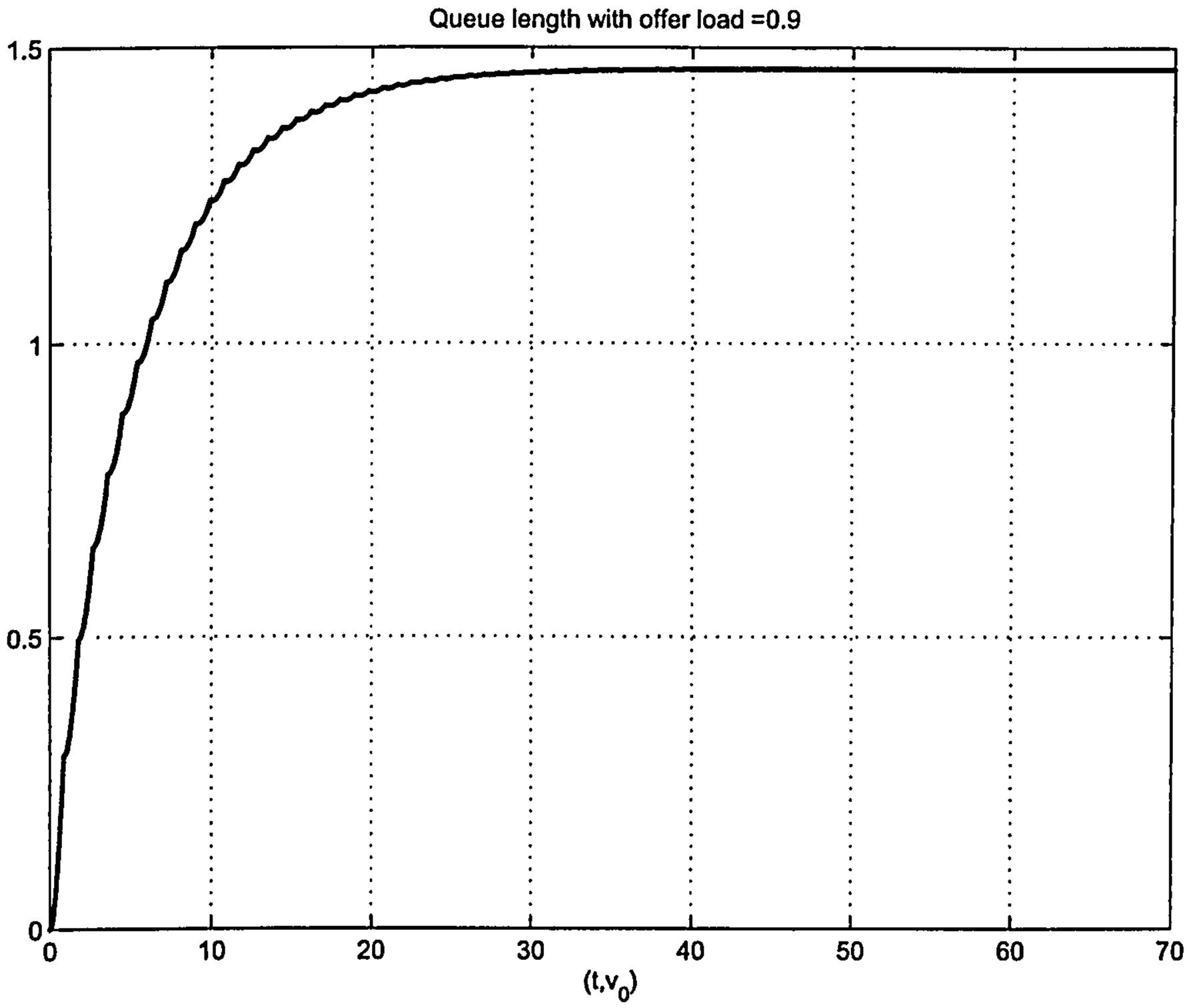


Figure 6.10: Que length with offer load $\rho = 0.9$

1. pick an **arrival** module, place it in the modeling environment. This is similar to the graphical interface in Matlab–SIMULINK environment. The arrival model prompts for the necessary arrival distributions. In $M/G/1/N$ queue we need an exponential distribution. In Arena an exponential distribution is specified with respect to its mean β, with density

$$f(x) = \frac{1}{\beta} e^{-x/\beta} \tag{6.6.1}$$

for $x > 0$.

2. pick a **depart** module and choose a general distribution. The Erlang $E_k(x)$ distribution[4] is defined with mean $k\beta$, having density

$$f(x) = \beta^{-k} x^{k-1} e^{-x/\beta}/(k-1)! \tag{6.6.2}$$

[4]Notice our code using Matlab we use Erlang differently from the way it is coded in Arena. We use the concept of k hypothetical stages where each stage has a mean holding time of $1/k\mu$, or exponential rate $k\mu$, so that the total holding time is μ or total rate is μ.

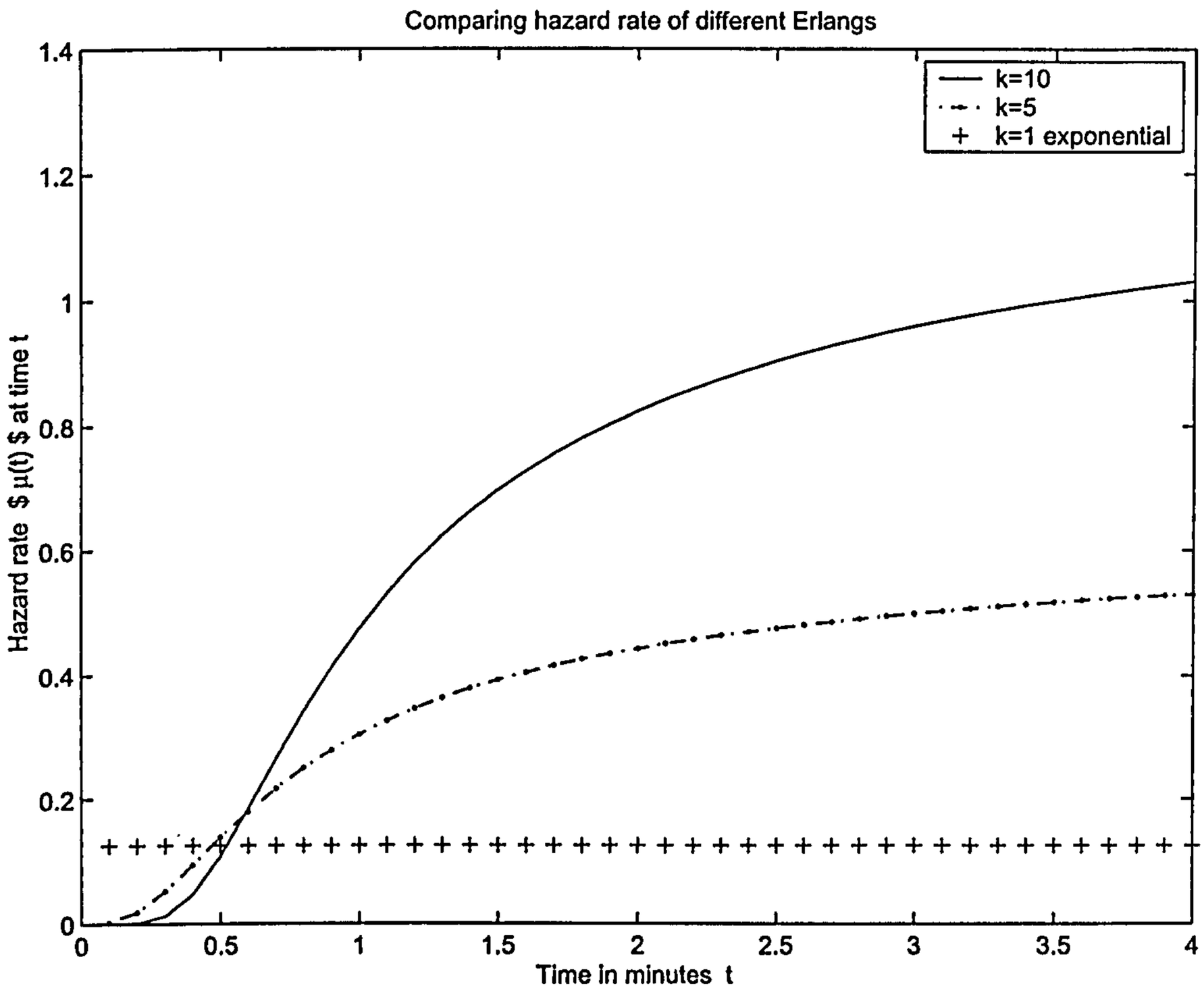


Figure 6.11: Hazard rates of some Erlang distributions

for $x > 0$.

3. choose the *server* module. In the server module from the Arena template, one is asked for such information as how long entities spend at the server, the server's operating schedule, and where entities should go.

4. choose a *simulation* module. This module provides facilities for choosing things like run length and number of replications.

5. choose the statistics module. This provides facilities for data acquisition, data analysis, computation of confidence intervals and plotting.

Animation is automatically included with many of the modules in the Arena template to allow for rapid development of both a simulation model and accompanying animation. Graphics symbols that are automatically provided when placing a module from the Arena.

Arena includes various animation options for real–time display of model statistics. For example, the user can place dynamic plots, histograms, levels, and time clocks directly within a simulation in order to illustrate system status as the model

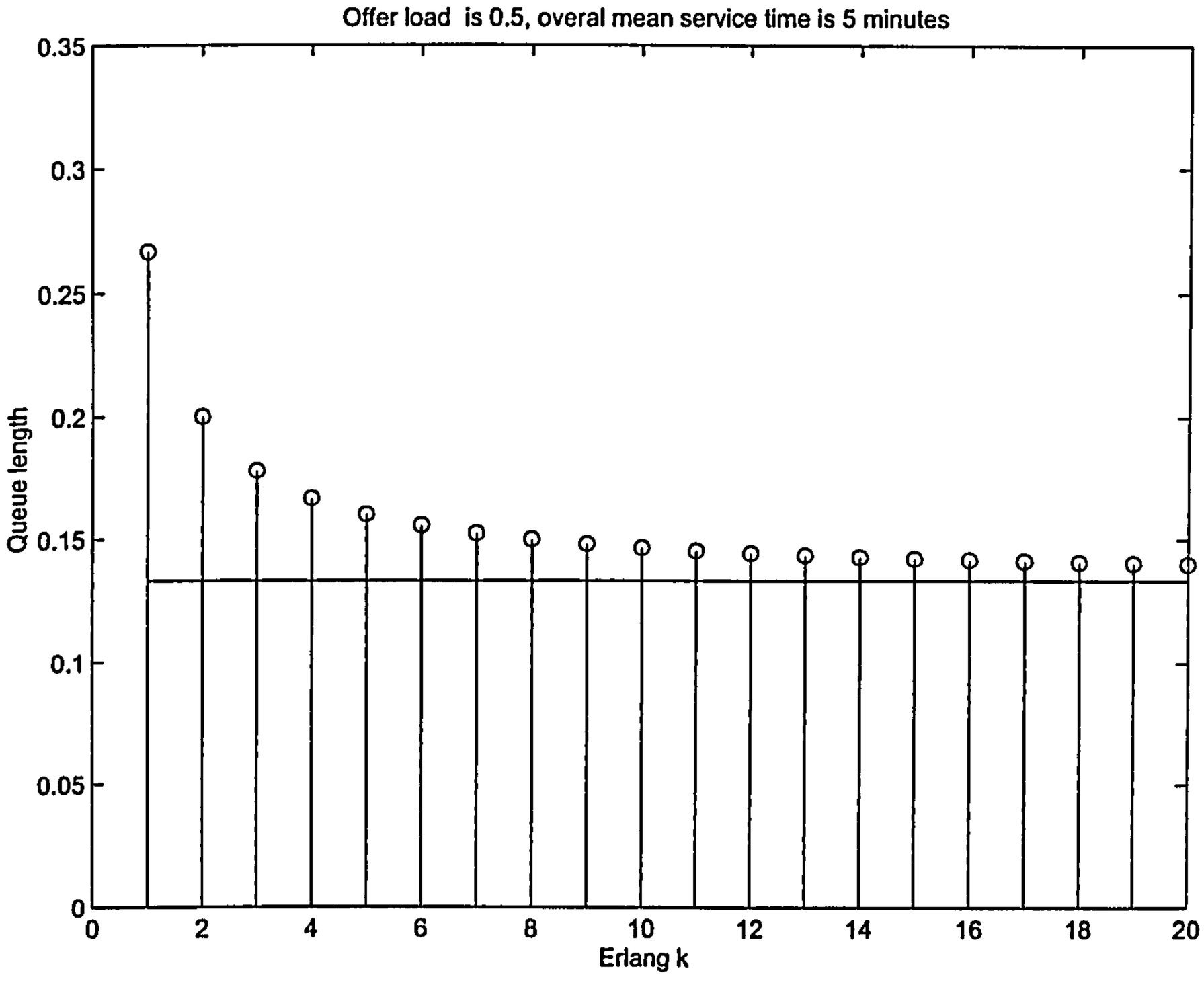


Figure 6.12: Variation of queue length with Erlang k

performs. This information is displayed on a real-time basis as well as on a post-process basis in the Arena statistical summary report.

6.6.1 SIMULIN–like flow diagram

Dating back some years ago, we have been experimenting with creating a data analysis modeling enviroment completely in SIMULINK. While, assessing the potential of Arena we have come to realise that it supports some of the features we are interested in developing toolkit for–to make creating simulation models an entirely graphical process. Nearly all the system behaviours are represented by using the graphical modules described above. Even logic such as IF/THEN/ELSE–type branching and queue selection rules are supported and the user can create a flowchart of a system by placing the appropriate graphical modules on the Arena simulation environment and directly connecting these modules. Non–animated modules from the Arena template that make up the majority of this flow–chart can be freely interspersed with modules that contain animation.

6.7 Software integration

Arena is a Microsoft Windows 95 and Windows NT product. The whole thing was developed using Microsoft's Foundation Classes (MFC) and is written in object–oriented Visual C++ language. This enables users to integrate with other tools. In fact we were able to make use of its compliance with microsoft to interface its output analysis module with Matlab running on window 95, with external data/application models and by making use of all of the standard user interface options such as toolbar buttons, function keys, etc. that are in use in all Microsoft Office tools.

6.8 Results of $M/G/1/N$ using PDPs theory

These are the results of the solution of the partial differential equations. The function call to the Matlab code developed is

$$que(\lambda, \mu_0, u, D, N, \phi_0, T, M),$$

where

- λ is the Poisson rate.
- μ_0 is Erlang rate with mean holding time $(1/\mu_0)$.
- u is not used here, set equal to unity, but it is a bandwidth of a communication link as in chapter (8.
- D is the service threshold or truncation, set so large that the probability that a service time will exceed D is negligible.
- N is maximum number of customers in the queueing system.
- ϕ_0 is not used here, set equal to zero, but it is one of the control boundary conditions, in the optimal control of demand and supply. There it represents that maximum possible units of goods that can be stored in the store or warehouse at a given time.
- T is the terminal time, set so that the transient effects is negligible at that time. The setting of T is problem dependent and it is not only dependent on the offer load but also on the actual values of rates λ, μ_0 and the number of

Erlang stages k. We have noticed that two queues with the same offer load and different combinations of λ/μ_0 have the same stationary response but different transient responses. In fact their responses follow different paths initially but converges to the same solution in the long run at T, for a good choice of T.

- M is the number of discrete intervals in the τ axis or x axis.

Here are the results of experiments using the function call *que*. the results show that queue length at time T when it is assumed that the transient effects have died out.

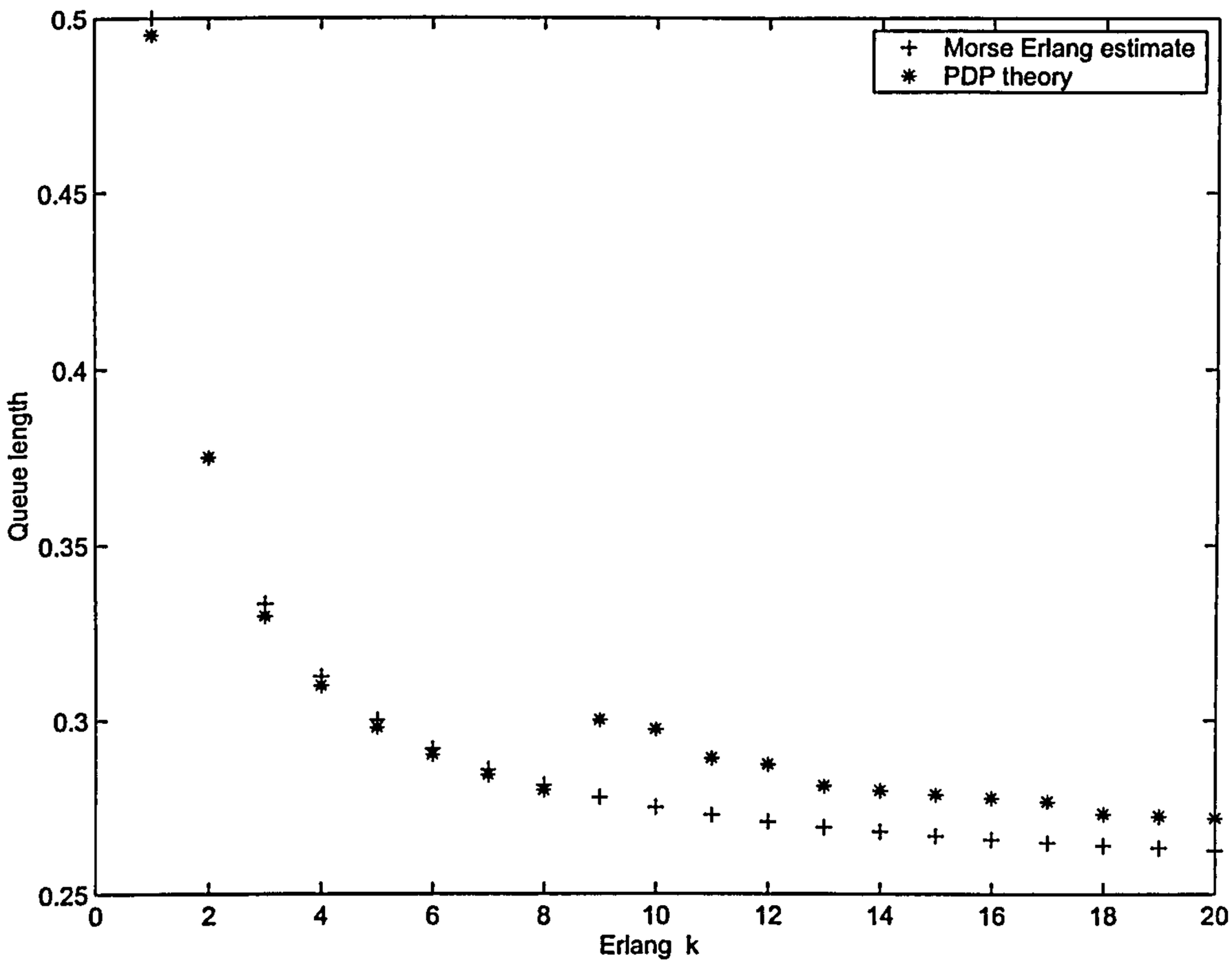


Figure 6.13: Comparing PDP with Morse's results

6.8.1 Comparism with Morse Erlang model

Philip Morse (Morse 1958) was a very distinguished Professor of mathematical physics, who in his later years got interested in queueing theory. Like our approach in simulation, Morse developed most of his works in queueing systems in terms of Erlang distributions. He developed a stationary distribution model for Erlang

service distributions, providing excellent asymptotic approximations for extreme values parameters such as the offer traffic. Morse's version for modeling, shared, in various ways, by us finally culminating in producing similar results, as the two results indicate. In Morse's works on Erlangian system did not take on board, as we do today, the concepts of renewal theory and elapsed service times.

Parameters	No. of Erlang stages	Queue length at T	
$que(\lambda, \mu_0, u, D, N, \phi_0, T, M)$	k	PDP theory	Morse
$que(0.1, 5, 1, 50, 20, 0, 100, 100)$	$k = 1$	0.4952	0.5000
$que(0.1, 5, 1, 50, 20, 0, 100, 100)$	$k = 2$	0.3750	0.3750
$que(0.1, 5, 1, 50, 20, 0, 100, 100)$	$k = 3$	0.3297	0.3333
$que(0.1, 5, 1, 50, 20, 0, 100, 100)$	$k = 4$	0.3099	0.3125
$que(0.1, 5, 1, 50, 20, 0, 100, 100)$	$k = 5$	0.2979	0.3000
$que(0.1, 5, 1, 30, 20, 0, 100, 100)$	$k = 6$	0.2900	0.2917
$que(0.1, 5, 1, 30, 20, 0, 100, 100)$	$k = 7$	0.2842	0.2857
$que(0.1, 5, 1, 30, 20, 0, 100, 100)$	$k = 8$	0.2799	0.2813
$que(0.1, 5, 1, 30, 20, 0, 100, 100)$	$k = 9$	0.3001	0.2778
$que(0.1, 5, 1, 30, 20, 0, 100, 100)$	$k = 10$	0.2974	0.2750
$que(0.1, 5, 1, 20, 20, 0, 80, 80)$	$k = 11$	0.2891	0.2727
$que(0.1, 5, 1, 20, 20, 0, 80, 80)$	$k = 12$	0.2873	0.2708
$que(0.1, 5, 1, 15, 20, 0, 80, 80)$	$k = 13$	0.2810	0.2692
$que(0.1, 5, 1, 15, 20, 0, 80, 80)$	$k = 14$	0.2797	0.2679
$que(0.1, 5, 1, 15, 20, 0, 80, 80)$	$k = 15$	0.2785	0.2667
$que(0.1, 5, 1, 15, 20, 0, 80, 80)$	$k = 16$	0.2775	0.2656
$que(0.1, 5, 1, 15, 20, 0, 80, 80)$	$k = 17$	0.2766	0.2647
$que(0.1, 5, 1, 12, 20, 0, 80, 80)$	$k = 18$	0.2730	0.2639
$que(0.1, 5, 1, 12, 20, 0, 80, 80)$	$k = 19$	0.2723	0.2632
$que(0.1, 5, 1, 12, 20, 0, 80, 80)$	$k = 20$	0.2717	0.2625

However, his thoughts predates the development of renewal theory that later followed (Cox 1966). We can deduce the following results:

1. As the Erlang stage k increases the mean queue length the mean queue length increases and this decrease in queue length occurs until convergence to an equilibrium value of that of $M/D/1/N$ at $k \to \infty$. This results confirm what is well–known that when the service becomes less random then the queue length becomes smaller. The highest queue length occurs when the service is exponential.

2. Note that as N is very large ($N = 20$) respect to the offer traffic $\rho = 0.5$ the

system behaves as an $M/G/1/\infty$ because the probability P_n of the buffer being full is almost negligible.

3. We have also found that as k decreases towards unity (the exponential case) then the truncation point needs to increase to obtain better estimation. This is because for a given mean–service–time the service density probability has a negligible tail as k increases, making it more likely that the service is complete for a given elapsed service time. Using simulation outputs we noticed the effects of k on the truncation D.

4. We also notice that the relaxation time, the time that it takes the system to settle down to its steady state is also a function of k and λ, and this is not just a function of the offer load ρ.

5.

6.9 Blocking probability

(Davis and Howl 1987) gave a method for computing the steady state probability for a $M/D/1/N$ but in that paper there did not give how to compute the blocking probability for a transient problem. Here we provide some results for the $M/G/1/N$ case. One method of doing this is to carry out the following procedure for a more general problem. Let $P_n(t)$ be the probability for the number of jobs in the queue at time t. Set the boundary conditions

$$\begin{aligned} V_k(\tau, T) &= 0, \text{ for } k \neq n, \\ V_n(\tau, T) &= 1, \text{ otherwise} \end{aligned} \tag{6.9.1}$$

and using the original generator for $M/G/1/N$, the backward partial differential equation in the Mayer form ($f_k(.) = 0$. $P_6(120) = 1.037e-9$ for $que(0.1, 5, 1, 10, 30, 0, 120, 120)$. The justification of this can be found in (Grigor'yan 1997) on the hitting probability for Brownian motion in terms of the Markov generator of the heat kernel. Figure (6.14) shows the hitting probability $P_6(t)$ for $que(0.1, 5, 1, 10, 30, 0, 120, 120)$. for $k = 3$.

Experiment 6.9.1 (Estimating the optimal value of D) *We need to estimate the value of D such that the tail of the service–time s distribution*

$$P(s > D)$$

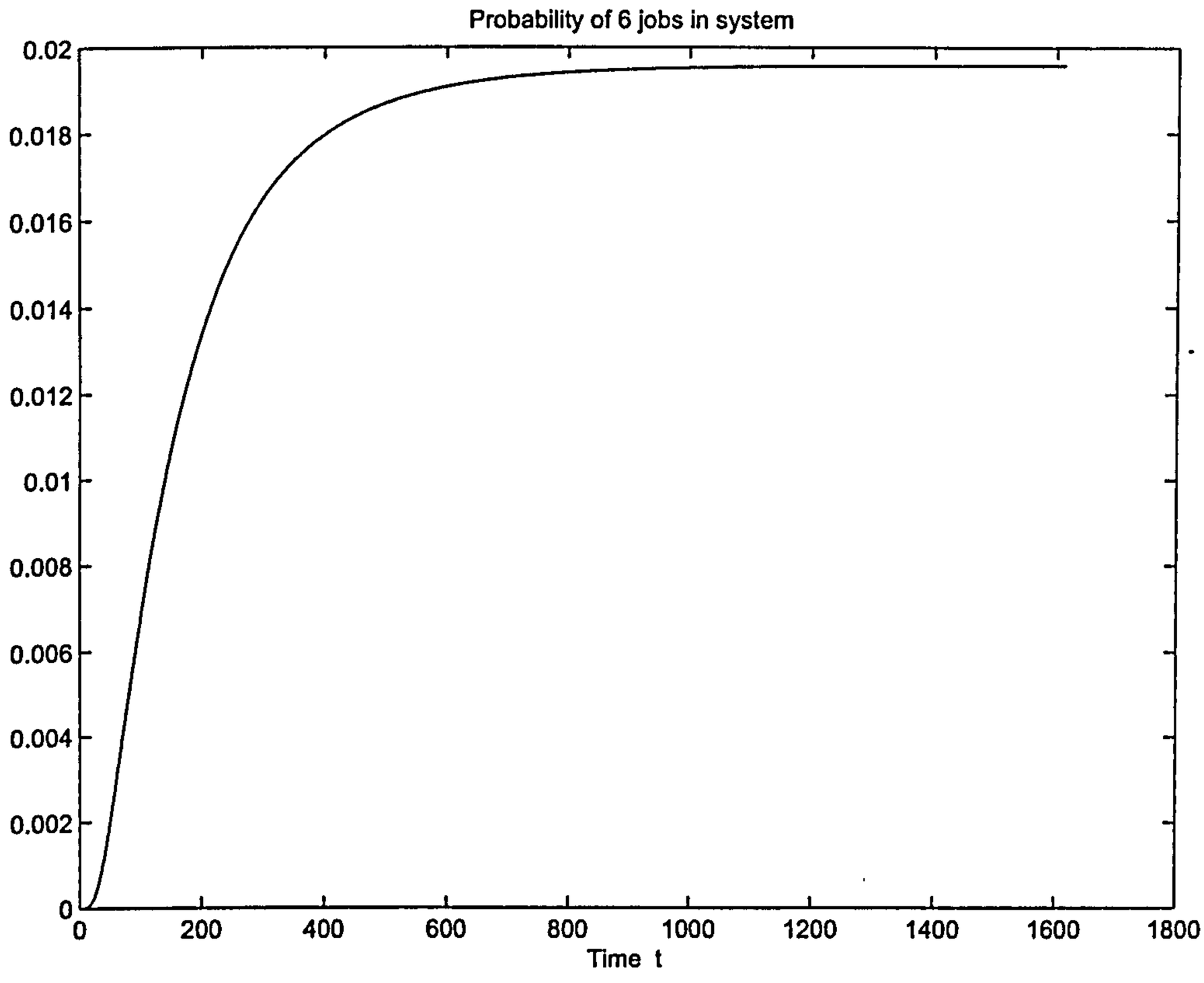


Figure 6.14: Hitting probability

is negligible.

We vary the values of D while keeping the other parameters constant. We found that the transient response of the system does not change for values of D above 20. But as D becomes small, say $7, 6, 5$ the response changes and converges to $(4.5, 4.3, 4.00)$ respectively. See

figure (6.19) for $D = 30$, figure (6.21) for $D = 20$, figure (6.20) for $D = 10$, figure (6.38) for $D = 7$, figure

(6.23) for $D = 6$, figure (6.25) for $D = 5$. figure (6.22) for $D = 4$.

Experiment 6.9.2 (Approximation for $M/D/1/N$)

To approximate of $M/D/1/N$, we set the hazard rate constant $\mu = 0$ and the results obtained in (Davis and Howl 1997) were reproduced.

Figure (6.17) shows the blocking probability μ_N with respect to the offered load. As expected μ_N increases with load. The idle probability is shown in figure (6.18), which decreases with offered load.

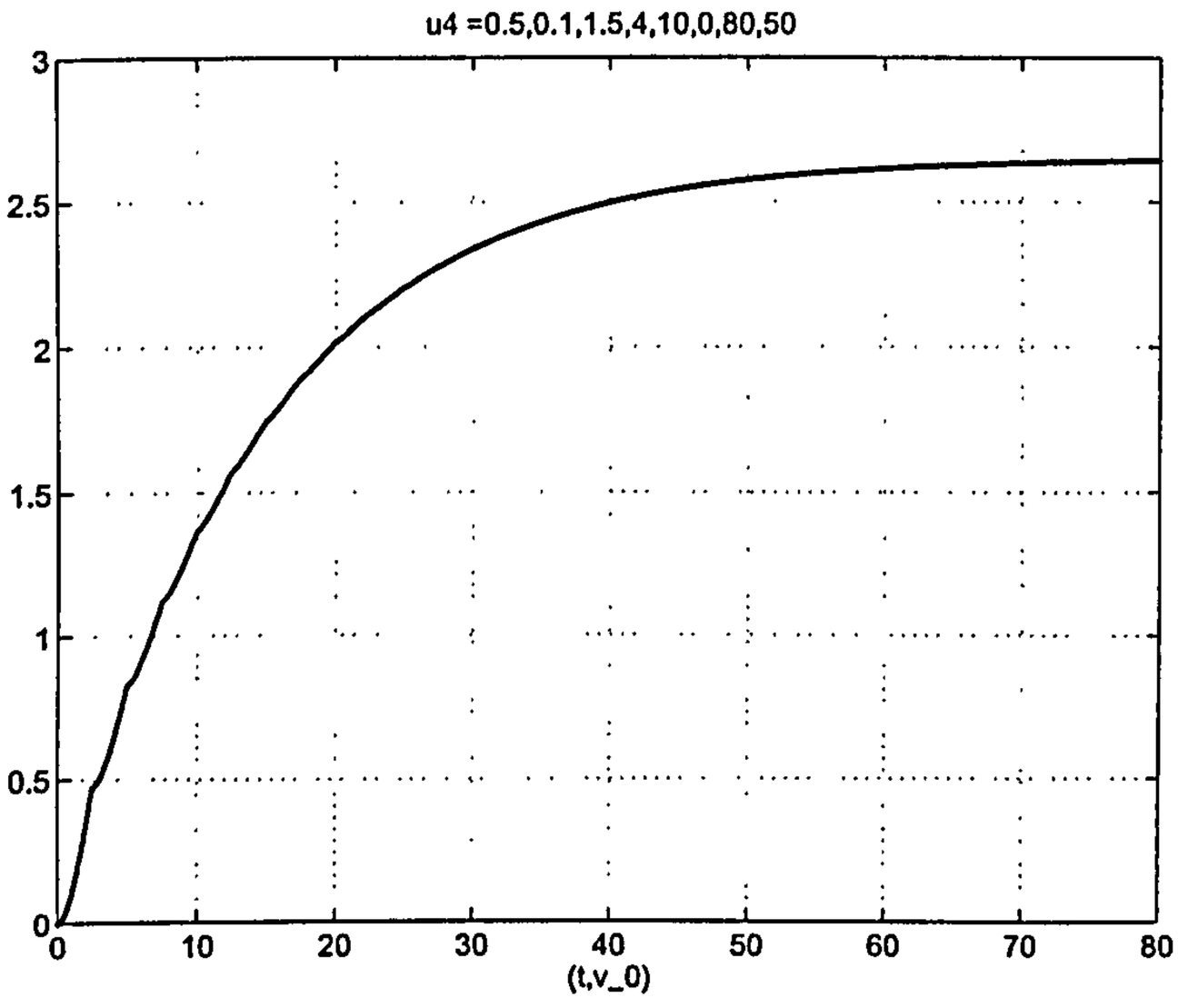


Figure 6.15: Davis queue

Figure (6.35) for $u = 6$, figure (6.34) for $u = 8$, figure (6.35) for $u = 4$. As shown in these figures the queue length decreases with rate increase in u.

6.10 Demand and supply

In the previous chapters (5) we wrote down the generator of $M/G/1/N$ in which was included the term $u_k(\tau)$ and $f_k(\tau, t)$. In the numerical analysis that followed the resulting partial differential equation we did not make use of these terms, but instead we set $u_k(\tau) = 1$ and $f_k(\tau, t) = 0$. We included these terms in anticipation of a class of control problems that uses various variations of an $M/G/1/N$ queue model. We started to study optimal control of queueing systems by looking at the capacity expansion problem of (Davis, Dempster, Sethi, and Vermes 1987), but it soon became apparent to us that the numerical solution of their model was based on an $M/D/1/N$ queue and what appeared to be needed that would capture a larger class of problems is the $M/G/1/N$. As it now turns out, the present formulation based on the $M/G/1/N$ is the first attempt to formulate the optimal control of queueing systems that incorporate a non–equilibrium feature and, by setting appropriate boundary conditions, the parameters $u_k(\tau, t)$ and $f_k(\tau, t)$, provide us with ample opportunities to apply this theory to:

1. Demand and supply: The main distinguishing feature of demand and supply

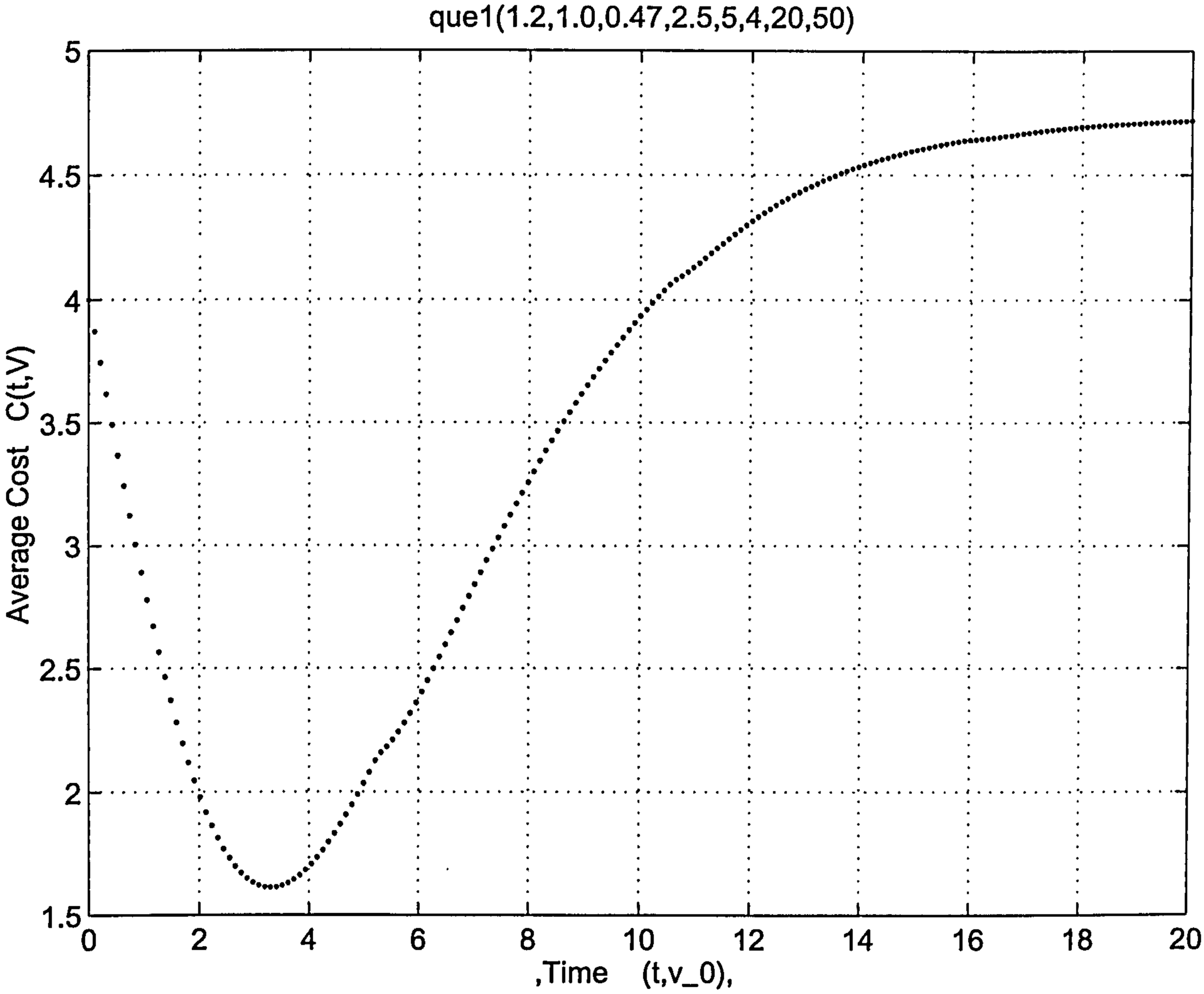


Figure 6.16: From Davis theory

processes which sets it off from the $M/G/1/N$ problem is that the "key " arrival, the individual customer causes a *decrease* from the queue length in our formulation not rather than an increase. When a service completion occurs, again from this view of things, the queue length is increase, caused by the outcome of the process of re–ordering or production of the goods.

2. Capacity expansion: As discussed in this chapter.

3. Insurance applications. The application of queueing systems to control problems in insurance is well know.

4. Telecommunication. Optimal control problems in telecommunication using queueing systems is well known.

The state of a capacity expansion can be defined as:

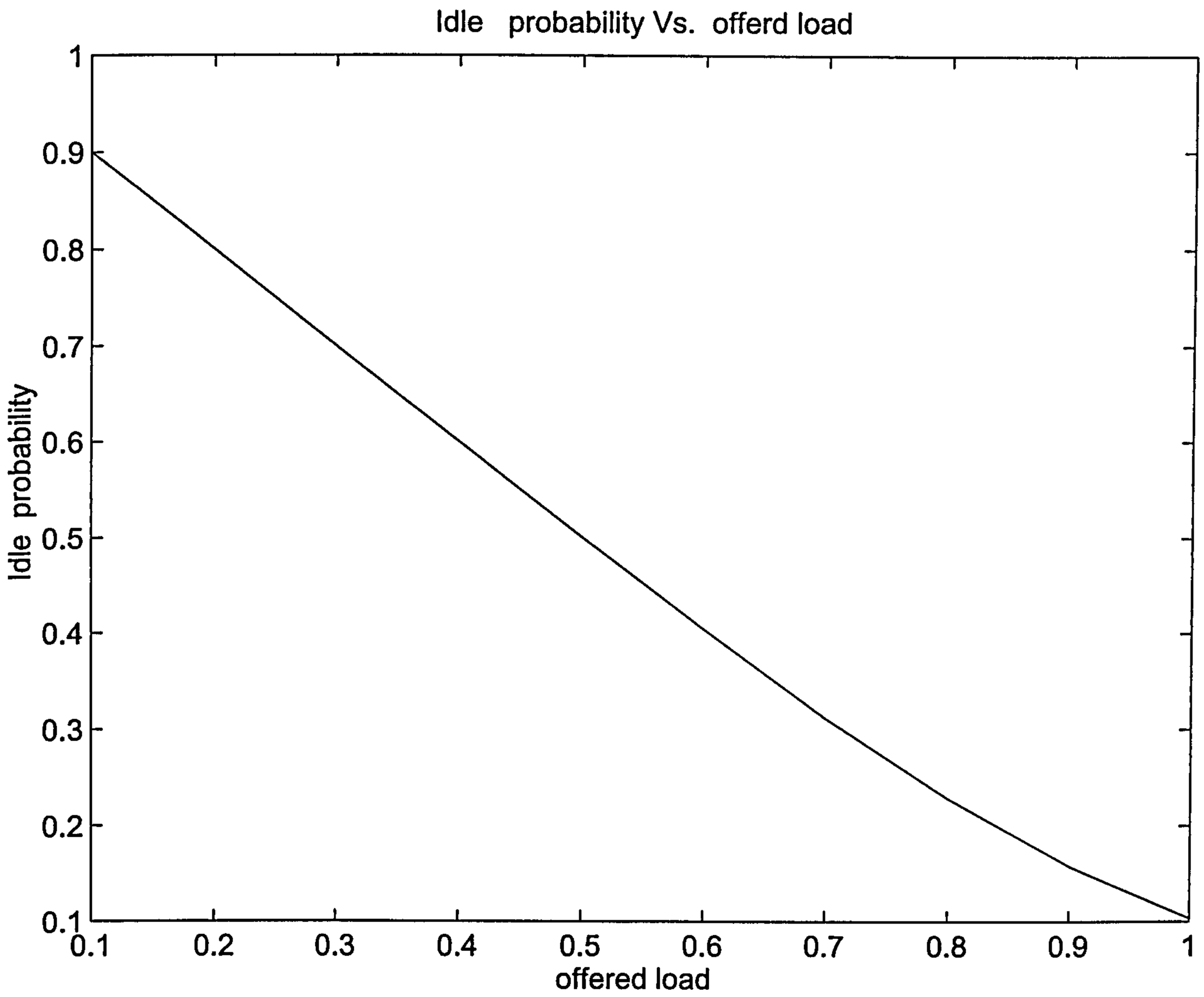


Figure 6.17: Blocking probability

$$V(\tau,t) = [V_{-M}(\tau,t), V_{-M+1}(t), V_{-M+2}(\tau,t), \ldots, V_{N1}(\tau,t)]^T, V(\tau,0) = \Phi(\tau),$$

where

$$\Phi(\tau) = [\phi_{-M}(\tau), \ldots, \phi_{N1}(\tau)]^T,$$

$$\phi_k(\tau) = |k-1|.$$

Negative k implies that there is excess capacity and positive corresponds to shortage of capacity. There are three possible cases:

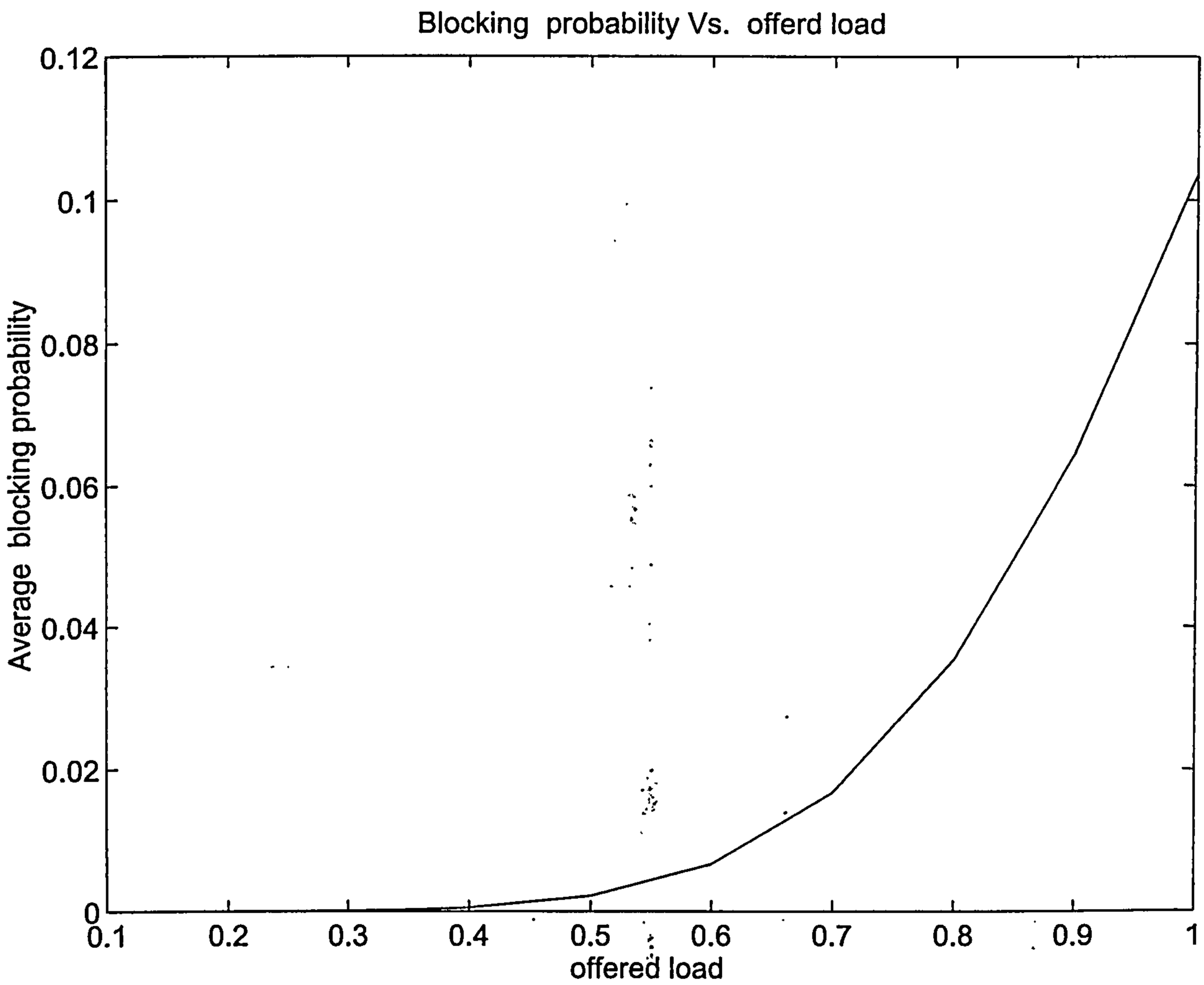


Figure 6.18: Idle probability

- **Case 1** $k = 0$: There are not customers and no excess capacity.
- **Case 2** $k > 0$: There are shortage of capacity, so that customers are waiting for a demand of k units of capacity.
- **Case 3** $k < 0$: There are excess capacity of k units and as soon as customers forward their demands of units d such that

  $(d < |k|)$ then there are satisfies immediately, leaving behind an excess capacity of $|k| - d$. But if the $(d > |k|)$ then there will be a shortage of $d - |k|$ units of capacity. It is assume that as soon as investment in constructing a new project is complete the capacity gets available immediately. This introduces the boundary condition

$$V_k(D, t) = V_{k-m}(0, t), \tag{6.10.1}$$

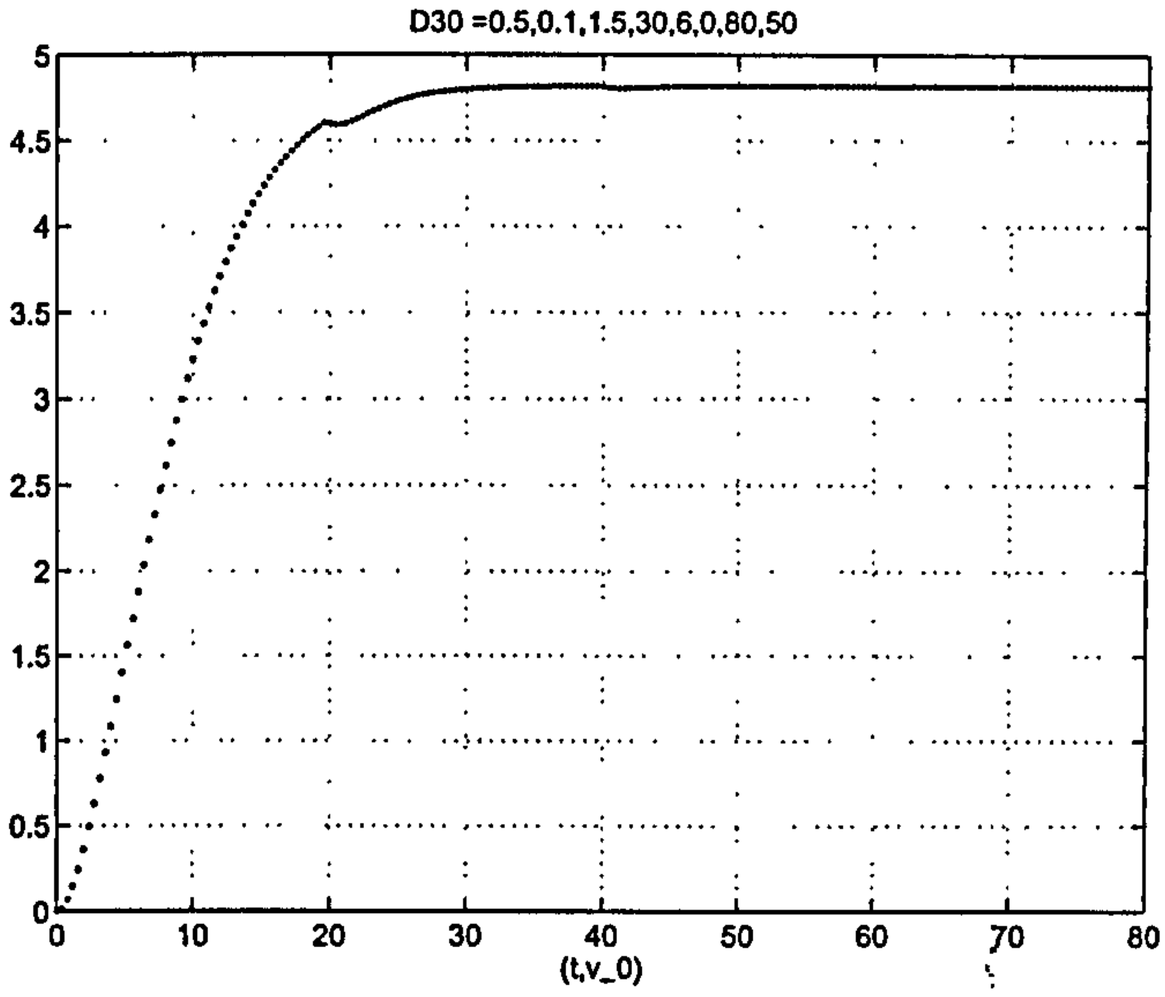


Figure 6.19: From Davis theory with D30

where m is the total number of units of capacity that becomes available from each project completion.

$N1$ is the maximum level of shortage and M is the maximum level of excess capacity.

By making the substitution $N = M + N1 + 1$ the state can be represented as

$$V(\tau, t) = [V_0(t), V_1(\tau, t), \ldots, V_N(\tau, t)]^T, V(\tau, 0) = \Phi(\tau),$$

where

$$\Phi(\tau) = [\phi_0(\tau), V_1(\tau), \ldots, \phi_N(\tau)]^T.$$

It is not hard to see that

Expansion equation

$$\frac{\partial}{\partial t}V(\tau, t) - u\frac{\partial}{\partial \tau}V(\tau, t) = A(\tau)V(\tau, t) + B(\tau)V(0, t), \qquad (6.10.2)$$

$$\begin{aligned} t &> 0, \\ \tau &\in [0, D], \end{aligned}$$

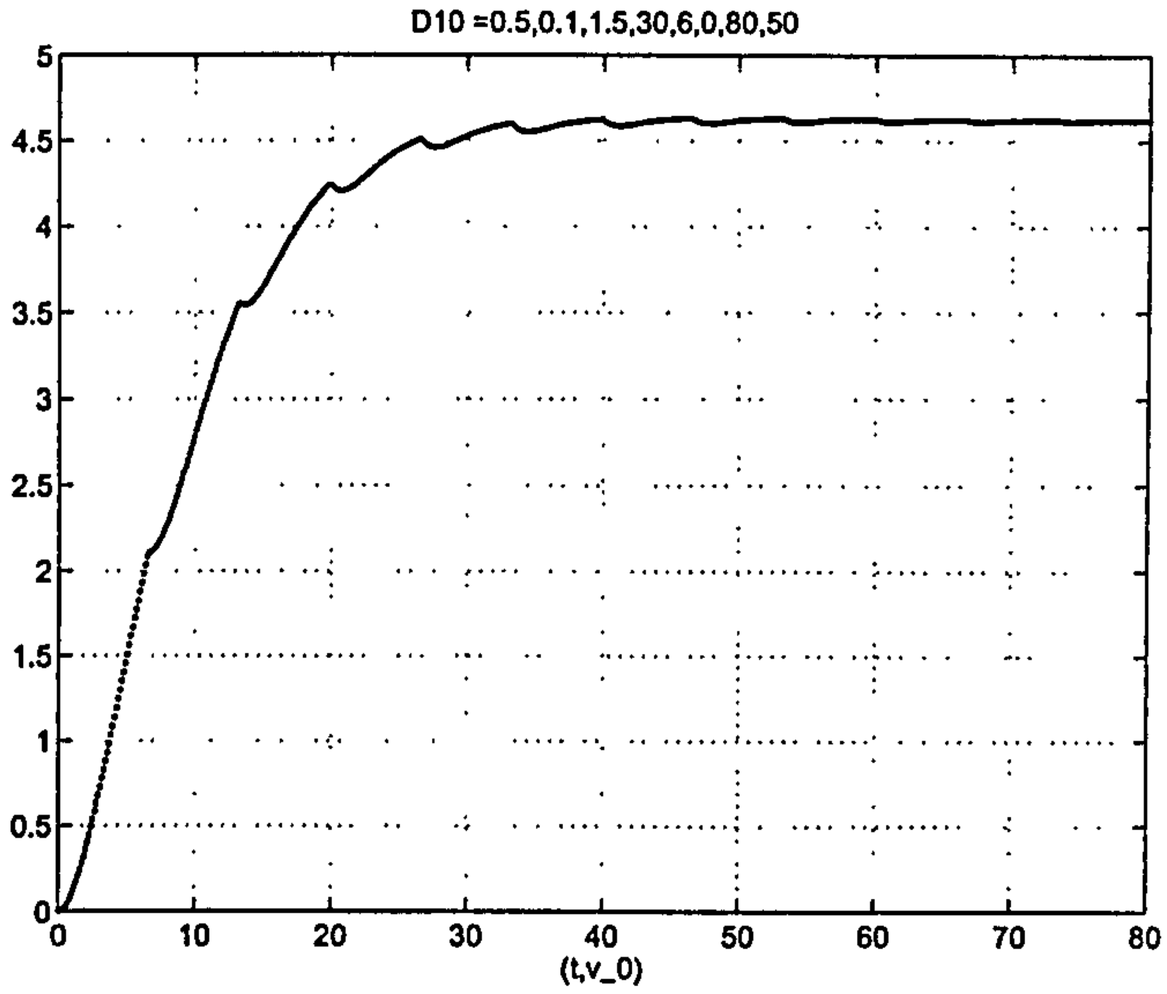


Figure 6.20: Davis theory with D10

with boundary condition

$$V(D,t) = CV(0,t)$$

where

$$A(\tau) = \begin{pmatrix} \alpha_0 & \beta_0 & \dots & 0 \\ 0 & \alpha_1 & \beta_1 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & \alpha_{N-1} & \beta_{N-1} \\ 0 & 0 & 0 & \alpha_N \end{pmatrix}. \tag{6.10.3}$$

The choice of B and C depends on the number of units m of capacity that are supplies, i.e. the number of customers that are served.

Without lost of generality, we shall assume that $m = 1$. Thus, B and C becomes

$$B(\tau) = \begin{pmatrix} \gamma_0 & 0 & \dots & 0 \\ 0 & \gamma_1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & \gamma_{N-1} & \beta_{N-1} \\ 0 & 0 & 0 & \gamma_N \end{pmatrix}, \tag{6.10.4}$$

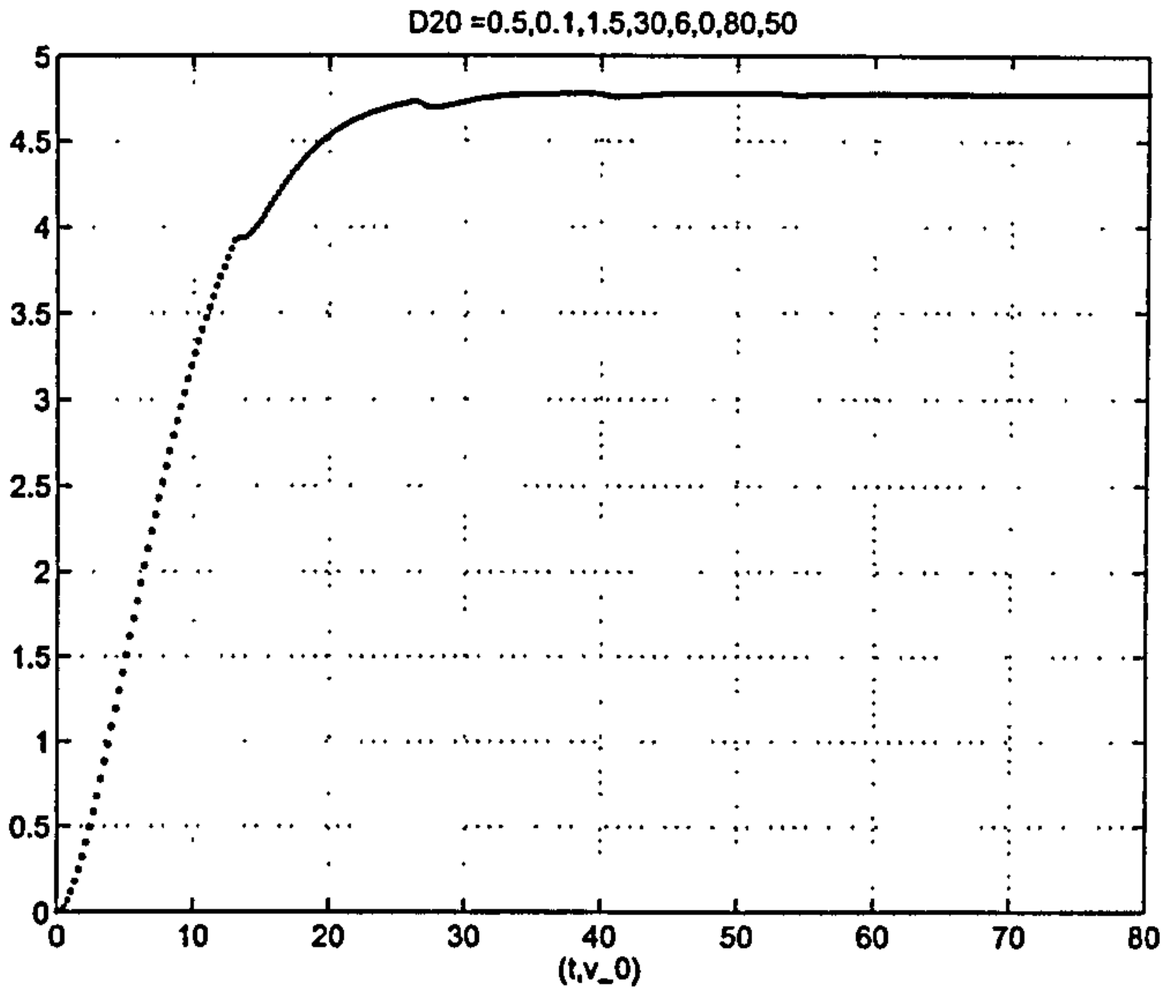


Figure 6.21: Queue length from Davis theory with D20

$$C = \begin{pmatrix} 0 & 0 & \dots & \dots \\ 1 & 0 & 0 & \dots \\ \vdots & \vdots & \vdots & \vdots \\ \dots & 1 & 0 & 0 \\ \dots & 0 & 1 & 0 \end{pmatrix}. \tag{6.10.5}$$

For the purpose of control, assume that the scalar u_k is a real value functions of (τ, t) for each k. Note, that, for a pair of given value of maximum excess capacity M and maximum shortage

level $N1$, the rate of investment is given as

$u_k(\tau, t)$ for $k \in [0, N]$. Thus, a sensible control policy must satisfy the inequalities

$$0 \leq u_1 \leq u_2 \dots, \leq u_N. \tag{6.10.6}$$

This implies that the rate of investment is greatest when the shortage level is greatest (i.e. when there is the highest demand in telecom services.)

With such control policies we shall have $N + 1$ characteristics c_i passing through each point in the control region of interest. From the theory of PDP, it is obvious that the slopes of these characteristics are given by $-u_k^{-1}$.

We assume that the coefficient matrices $A(\tau)$ and $B(\tau)$ and the control u_k are continuous and have continuous partial derivatives in the closed region $[0, D]$

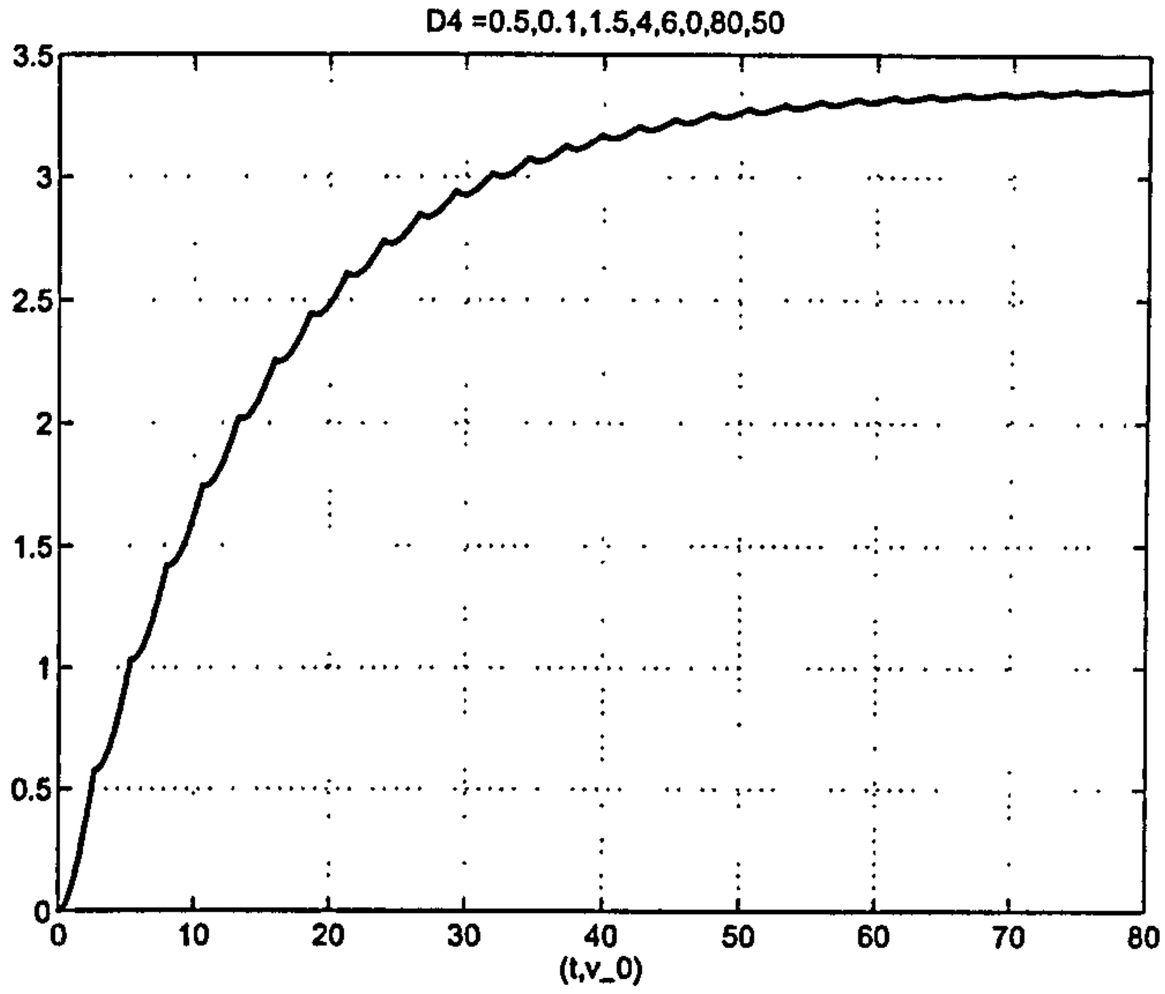


Figure 6.22: Another truncation D

Here we solve the equations and simulate it with respect to u. All of the parameters are kept constant and the investment rate changes in the following order: $u = 2, 4, 0.5, 0.6, 0.7, 0.2$.

Figure (6.26) for $u = 2$, figure (6.27) for $u = 0.4$, figure (6.28) for $u = 0.5$, figure (6.29) for $u = 0.7$, figure (6.30) for $u = 0.2$. As shown, the cost first decreases due to the fact that initially there is an increase in demand that reduces the excess capacity. This decrease in cost occurs for about 3.2 unit of time, after which, depending on the investment rate, two things occur: (1) the cost converges to a stationary cost for investment rate greater than 2 (2) the cost increases for investment rate that are less than 1.

Experiment 6.10.1 (Results for two–server case $M/G/2/N$)

Here the harzard rate of first server is 0.1 and the second server is 0.2. $N = 6$, $u = 1$, $D_1 = D_2 = 1$. The following results are obtained. See figures (6.36 and (6.37). This can be applied as in chapter(8) to study data communication through two adaptors on a switch.

6.11 Conclusion

The partial differential equations (PDEs) describing the dynamic of $M/G/1/N$ is the first of its kind to give a transient solution of a finite buffer queue with general

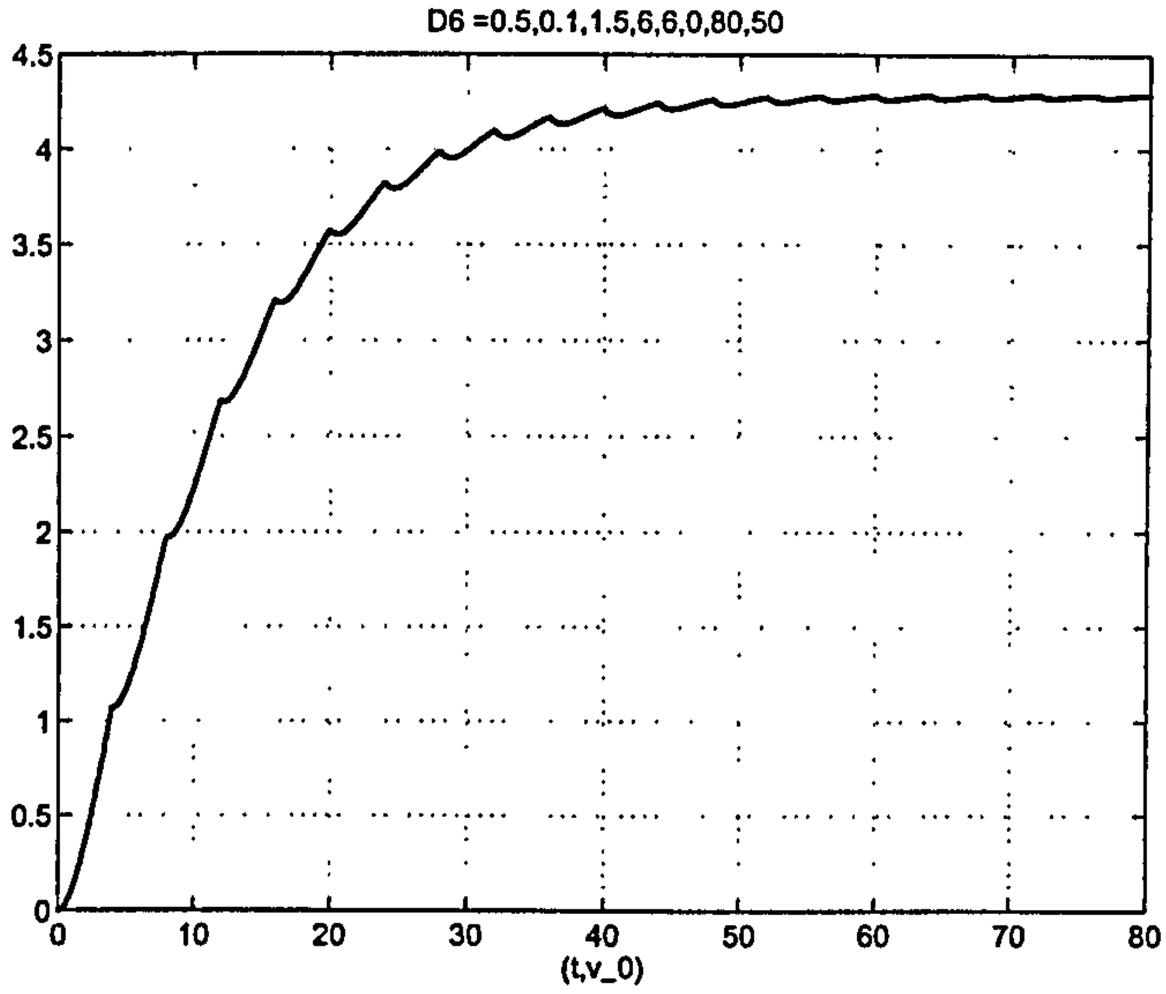


Figure 6.23: Effect of D

distribution. As far as we know the use of PDP theory in providing a computational model for transient queue is new. We extend this to the two server case and the derivation of the algorithm for computing queue length is made.

In this chapter we have done some comparisons the PDP queue model with output simulations. The results of these simulations as well as some portions of the results for computing steady state from simulations refer back to the work done in chapter(3). The validation of queue length distributions with simulations was suggested as a mean of testing the power of PDP model. Personnally, the comparison has shown that PDP theory is more closed related to queueing theory than had previously thought.

Secondly, the results of this chapter give rise other practical applications, ranging from demand and supply, capacity expansion and insurance to data communication.

With the present set up fairly good computation models are emerging, bringing the subject to the domain of computer science, with good consequences for communication systems. It is Bertrand Russell who once said that deep theories die not by refutation, as Sir Karl Popper spent his entire life believing, but by neglect. We hope the computational nature of this work will increase interest in the applications of PDP, averting Russell's greatest fear.

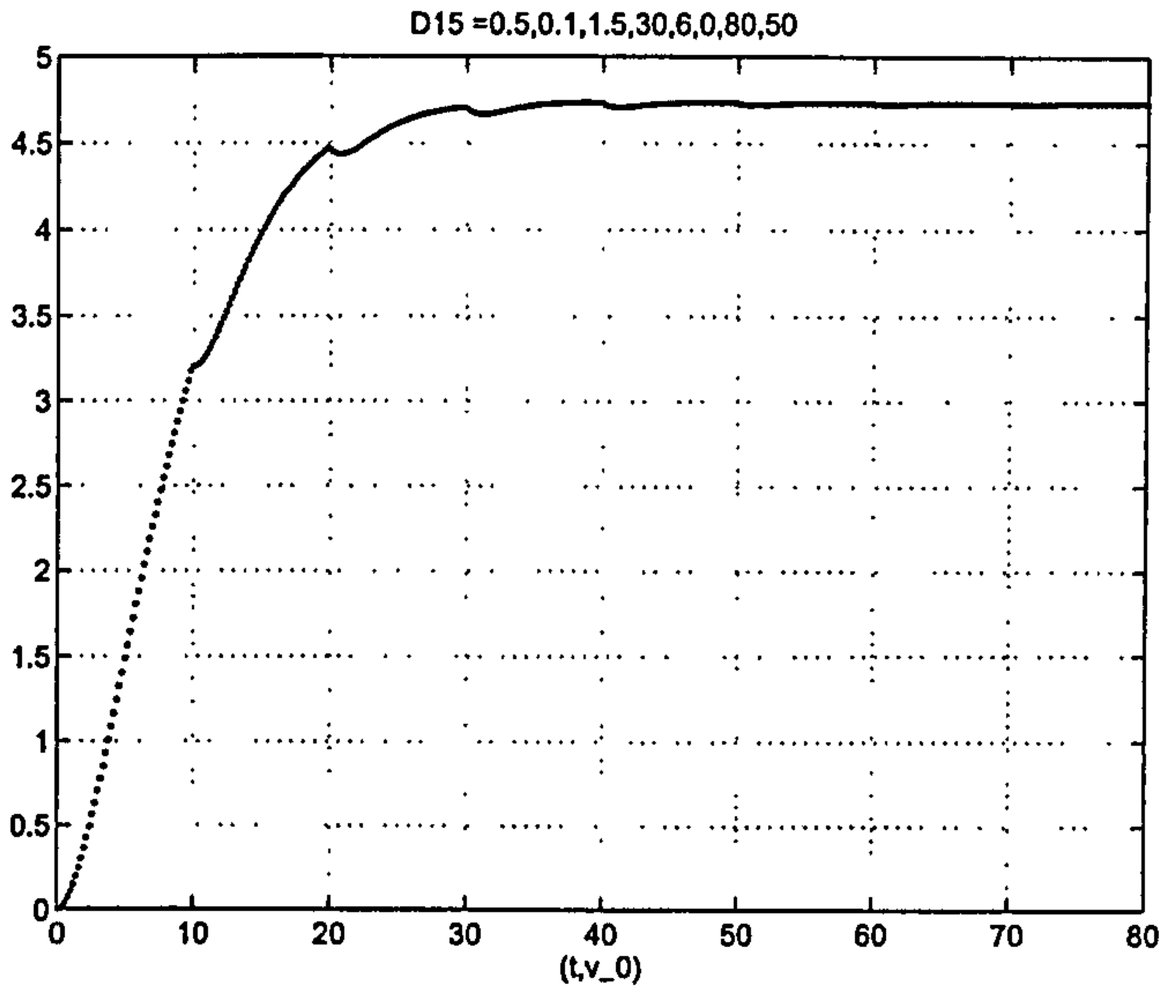


Figure 6.24: Truncation D

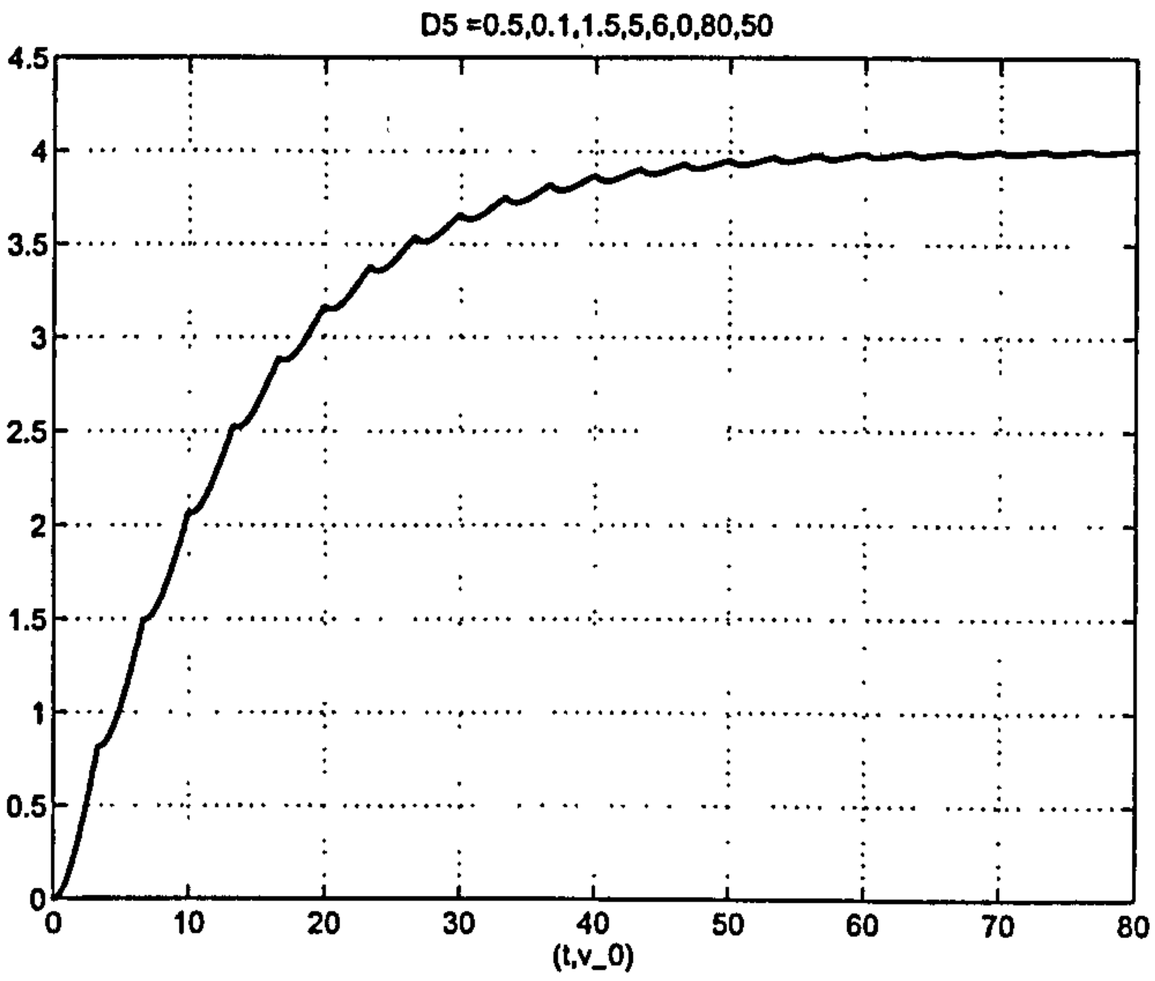


Figure 6.25: Effects of truncation D

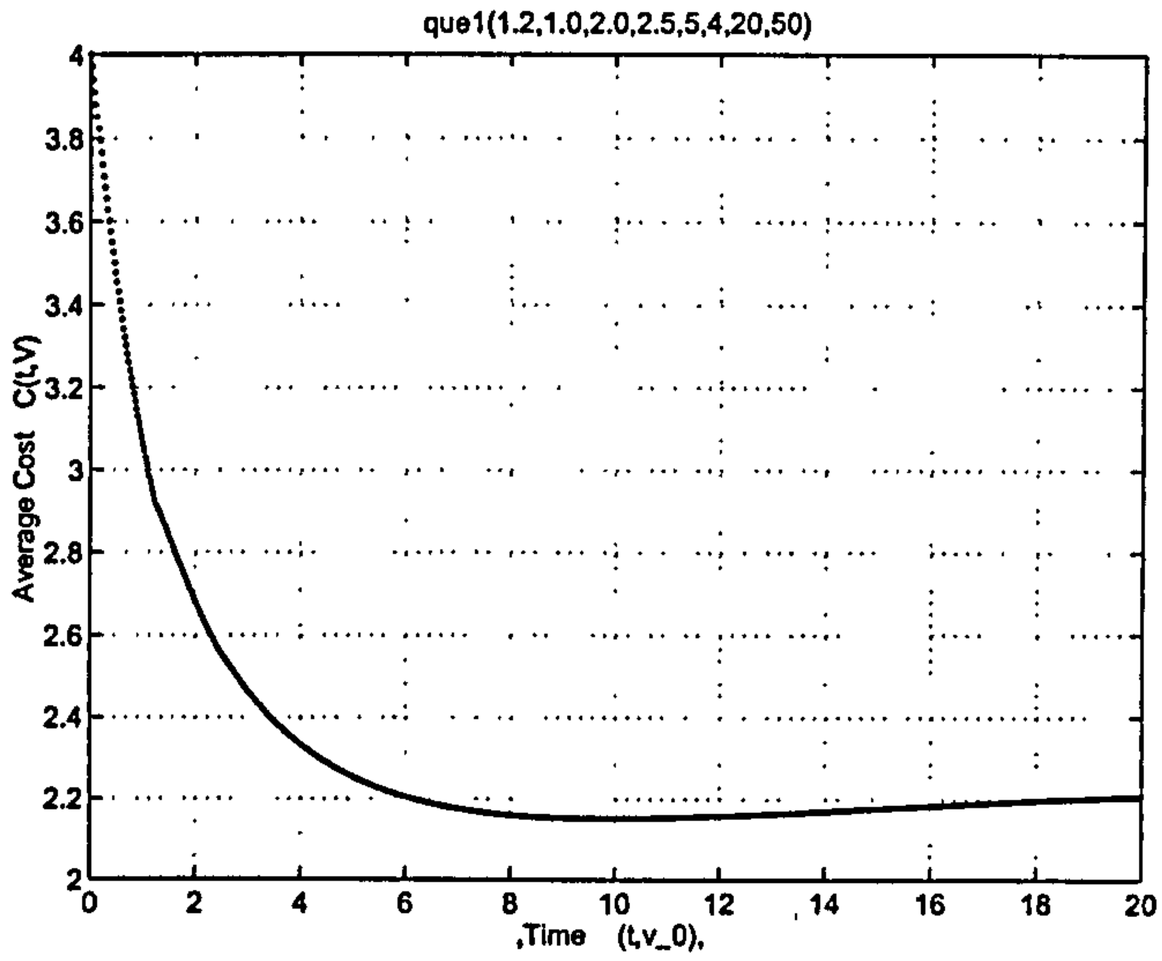


Figure 6.26: More queue transient

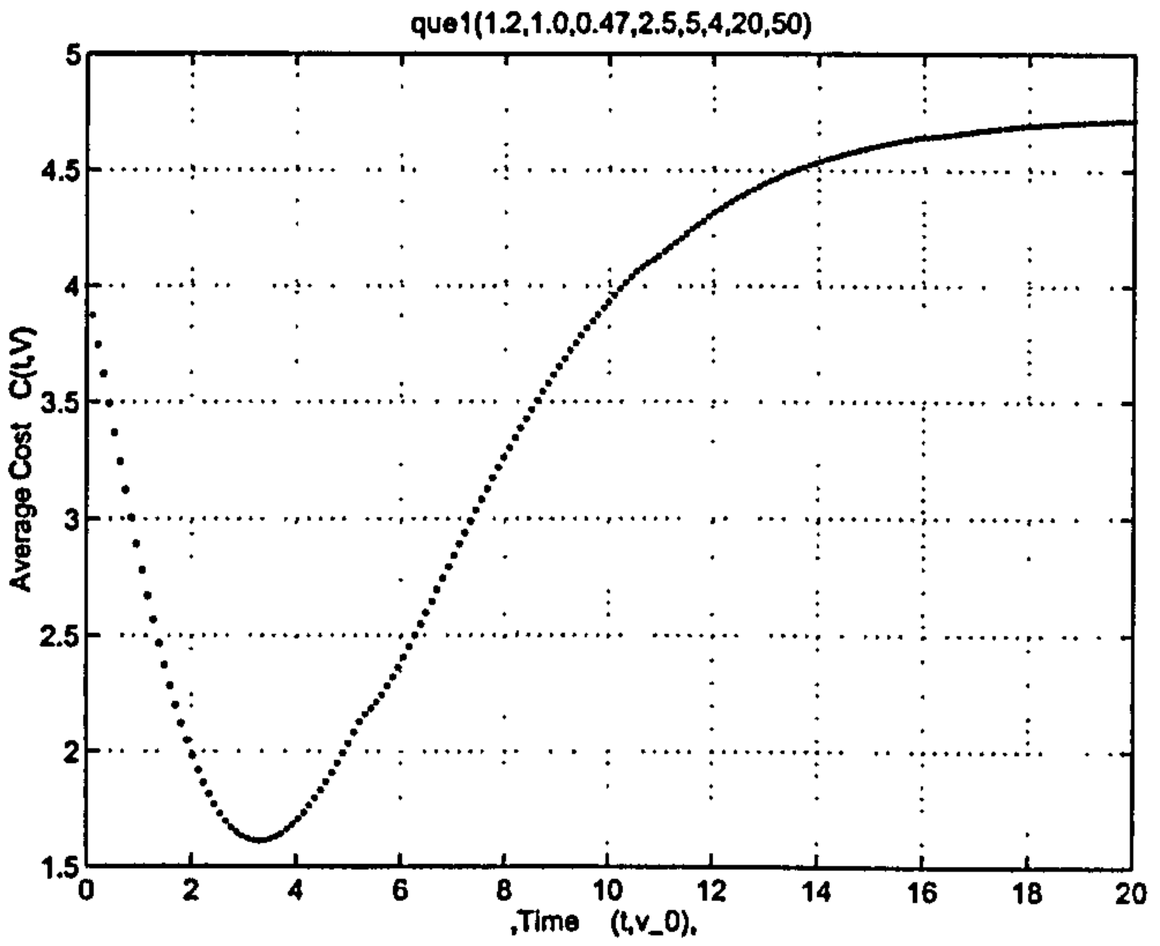


Figure 6.27: Queue example

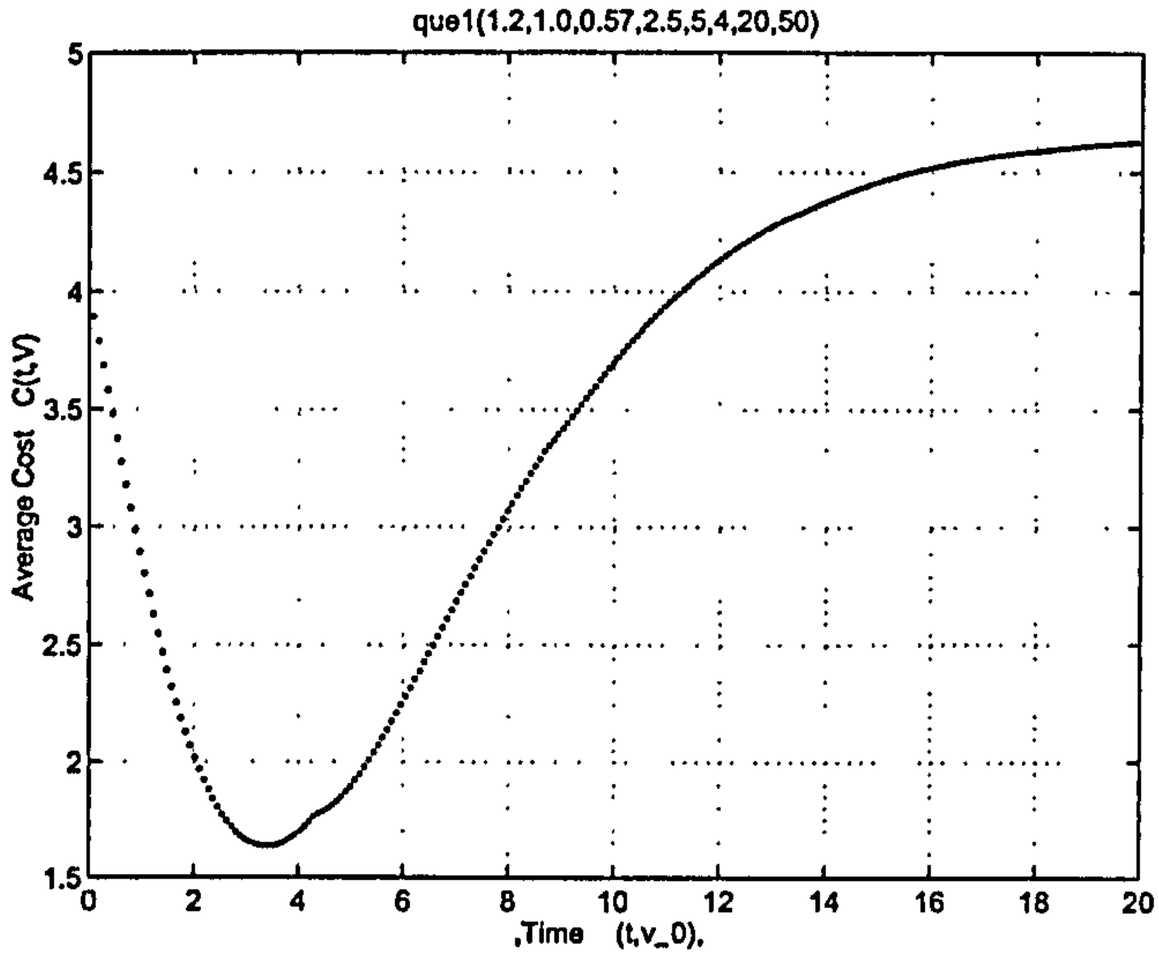


Figure 6.28: Queue example

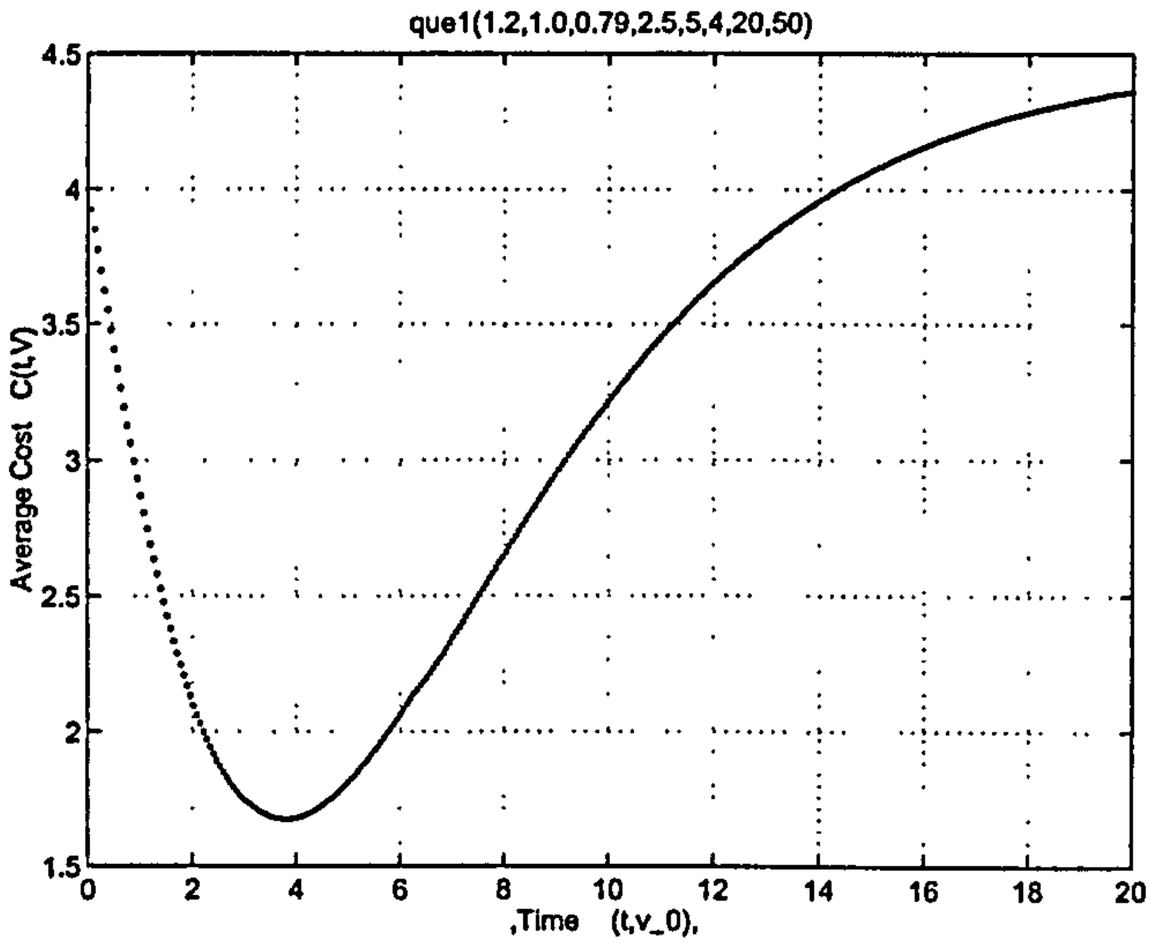


Figure 6.29: Example queue

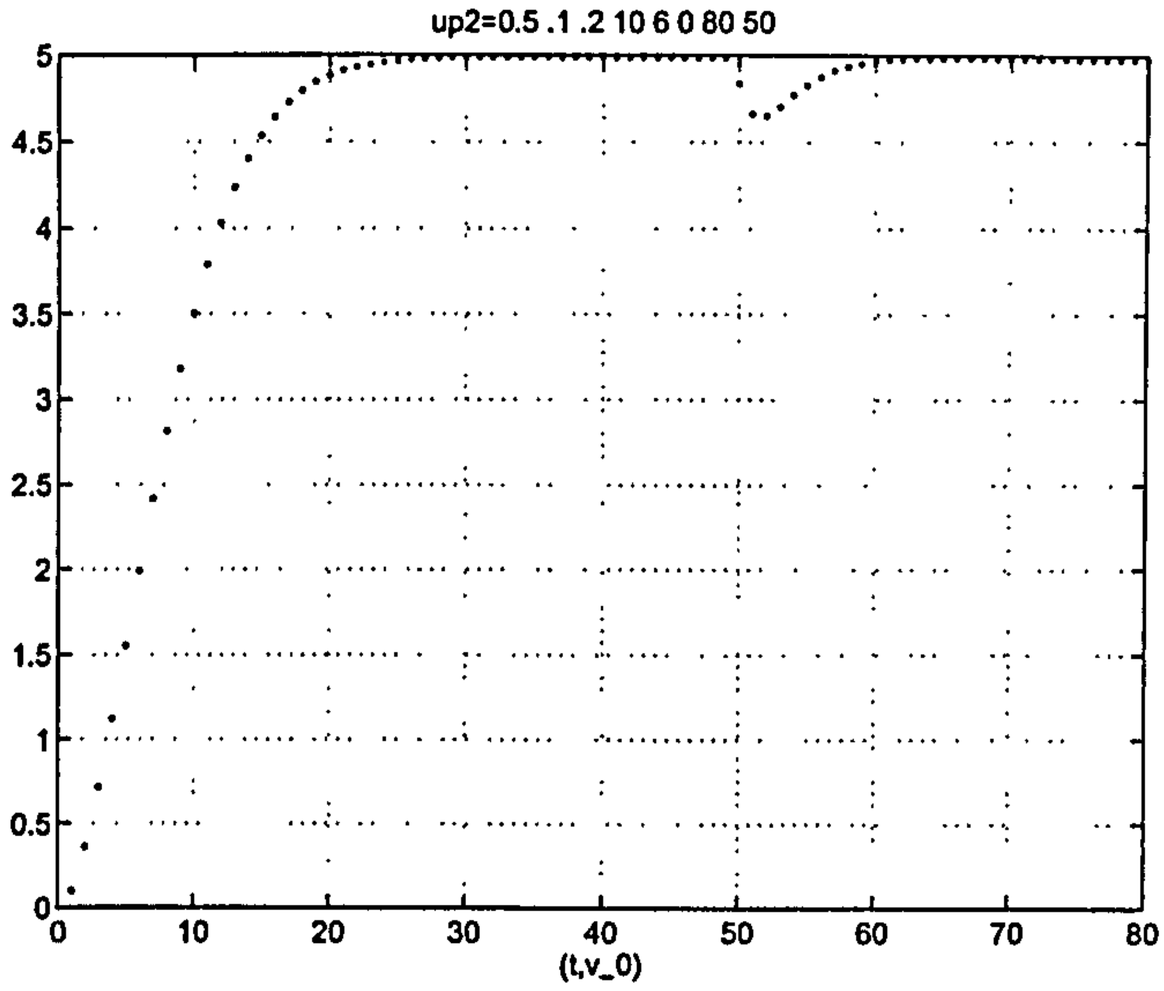


Figure 6.30: Queue illustration

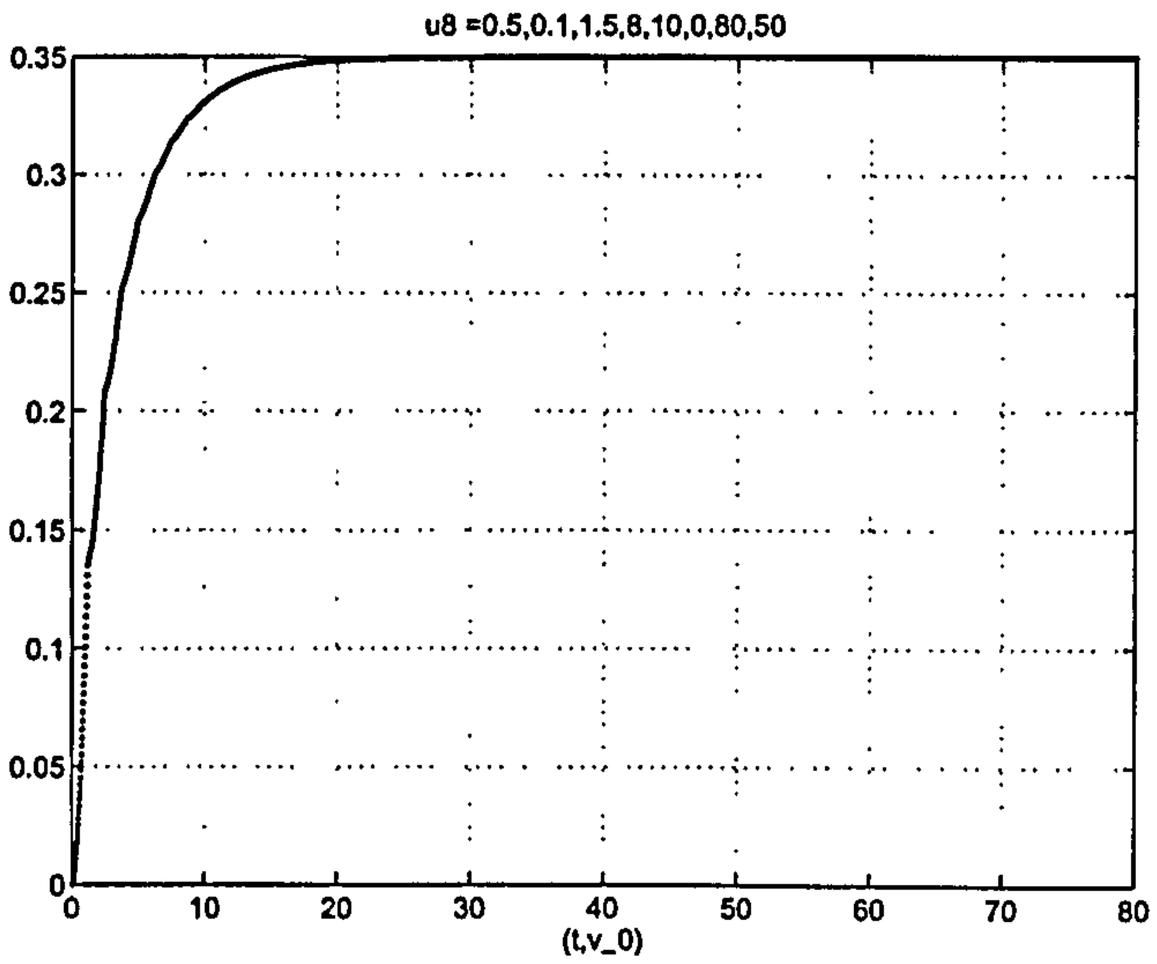


Figure 6.31: Example queue

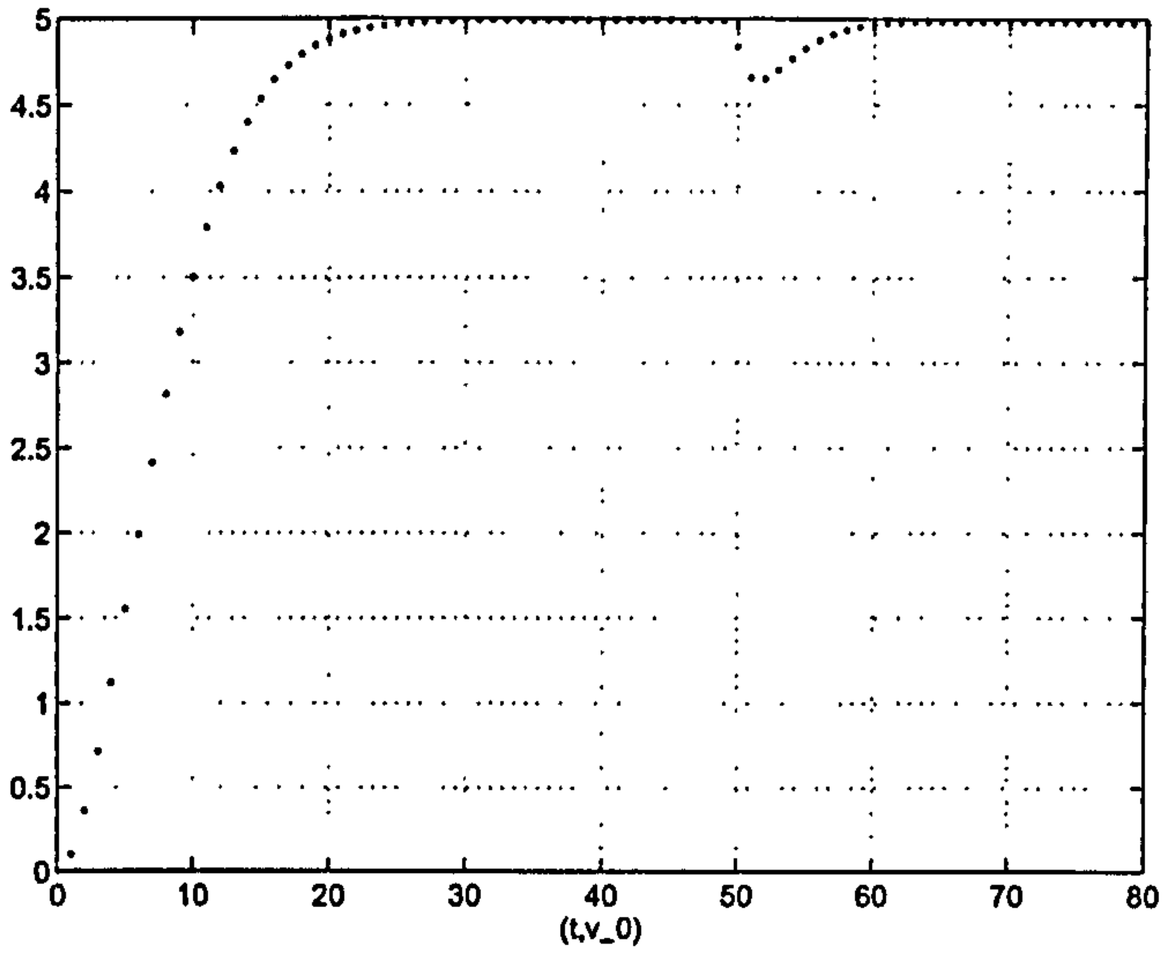


Figure 6.32: Queue with more speed of server

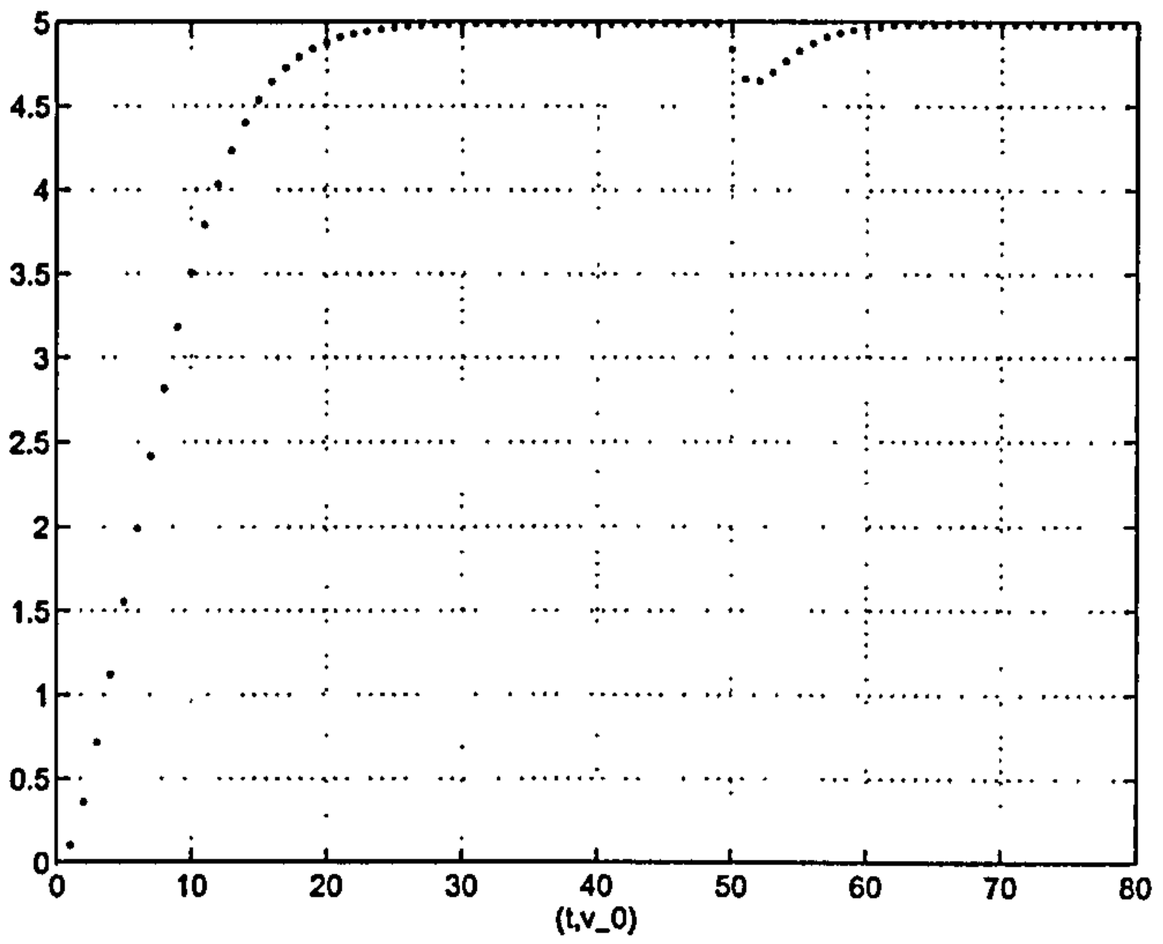


Figure 6.33: Effect of server speed

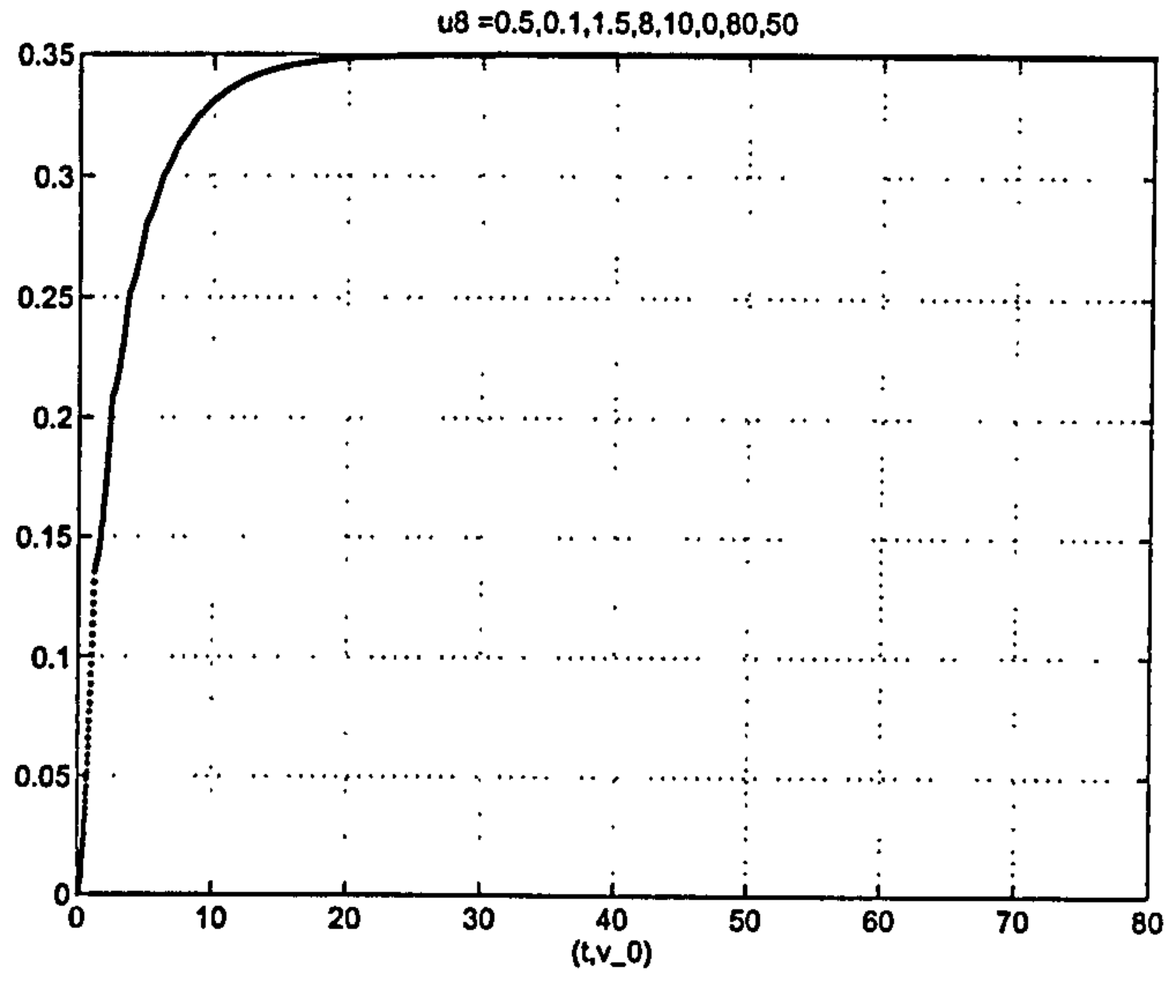


Figure 6.34: Another illustration of the effect of server speed

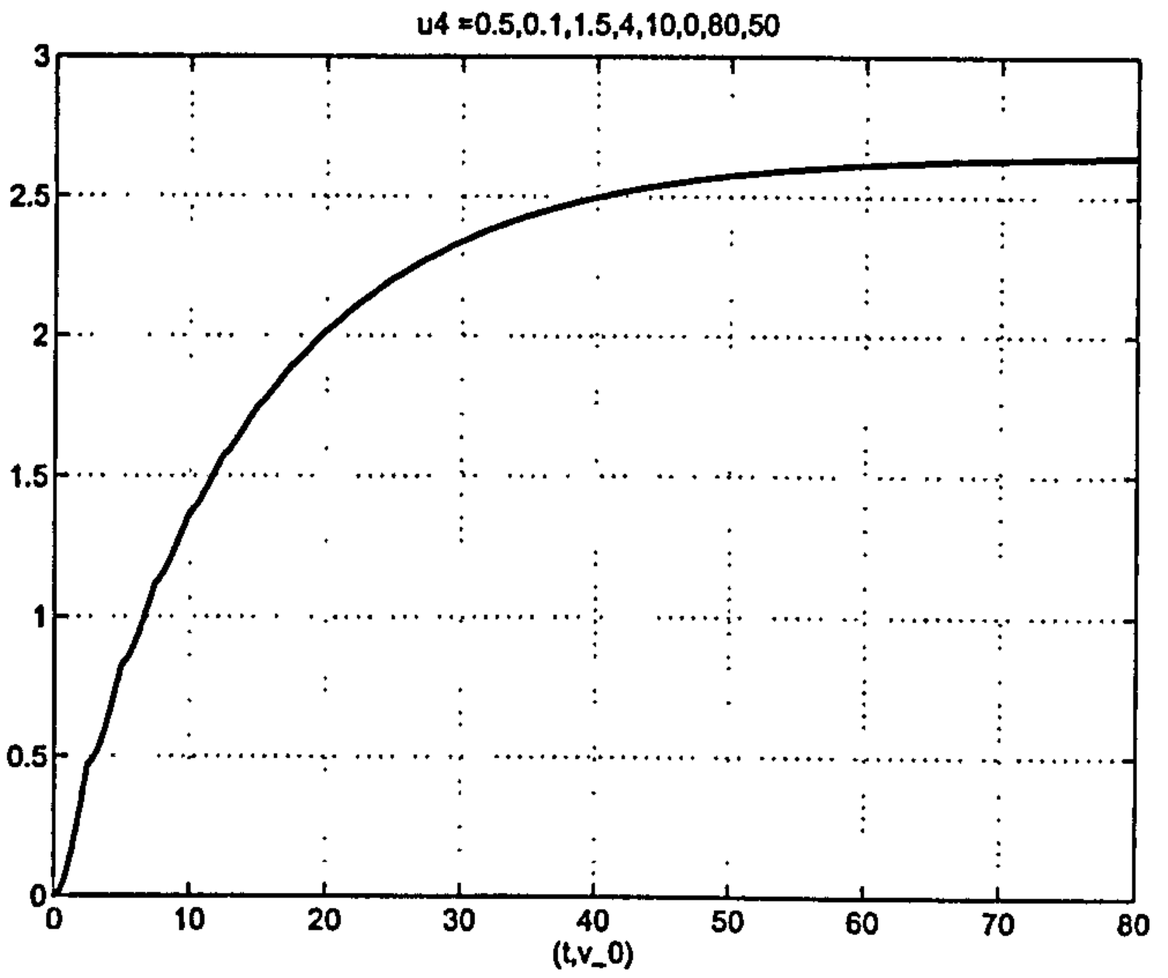


Figure 6.35: Queue with increase speed of server

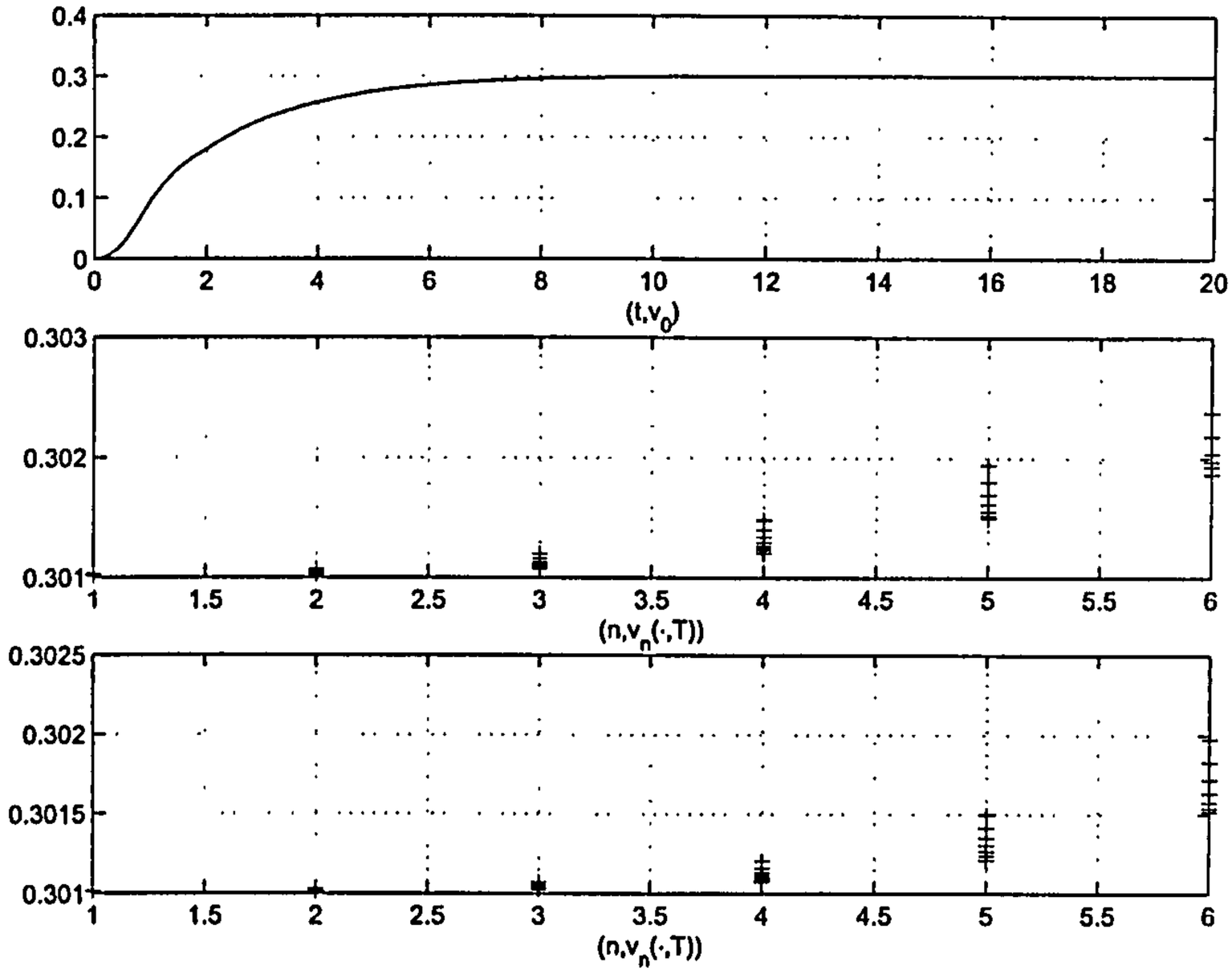


Figure 6.36: Speed of Second server is twice that of first server

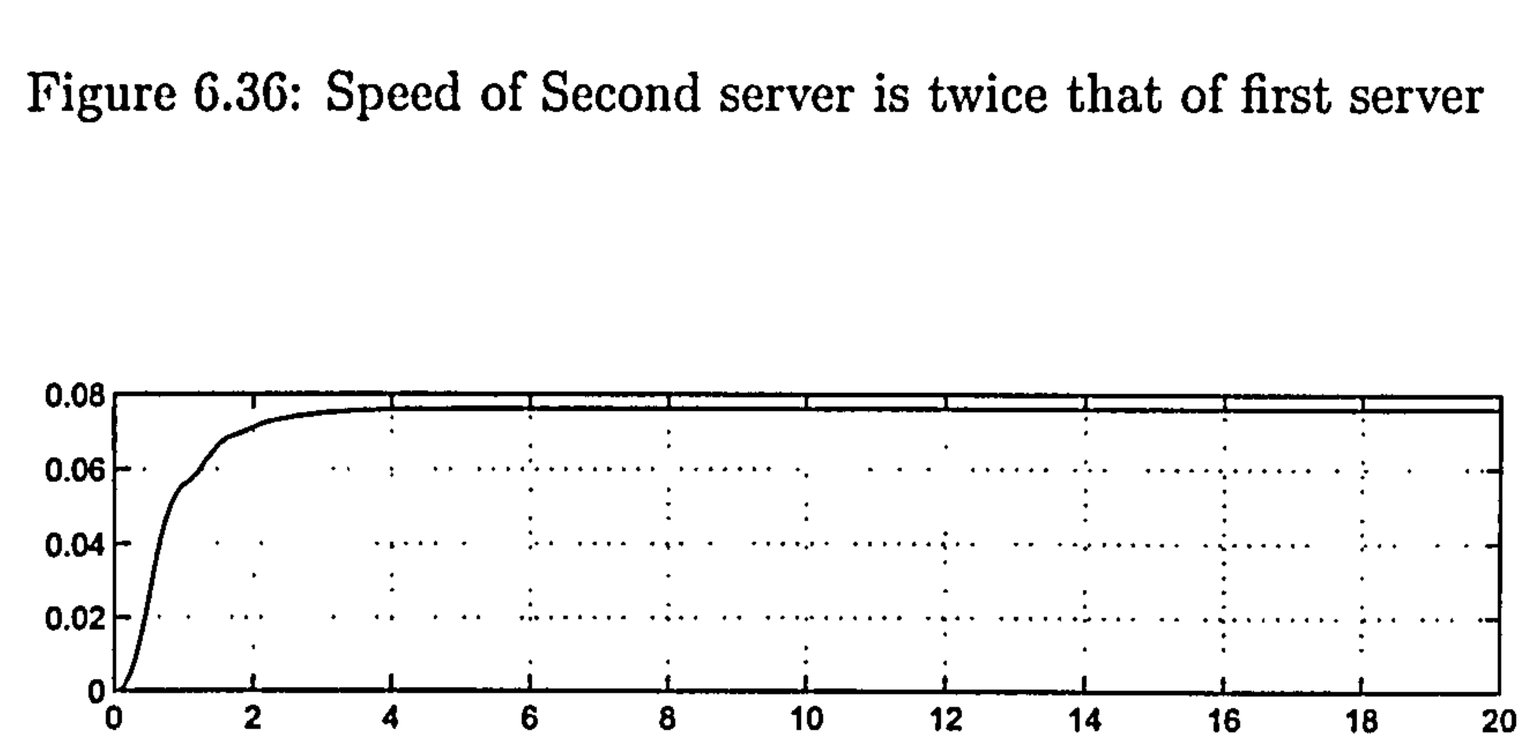


Figure 6.37: Speed of Second server is 3 times that of first server

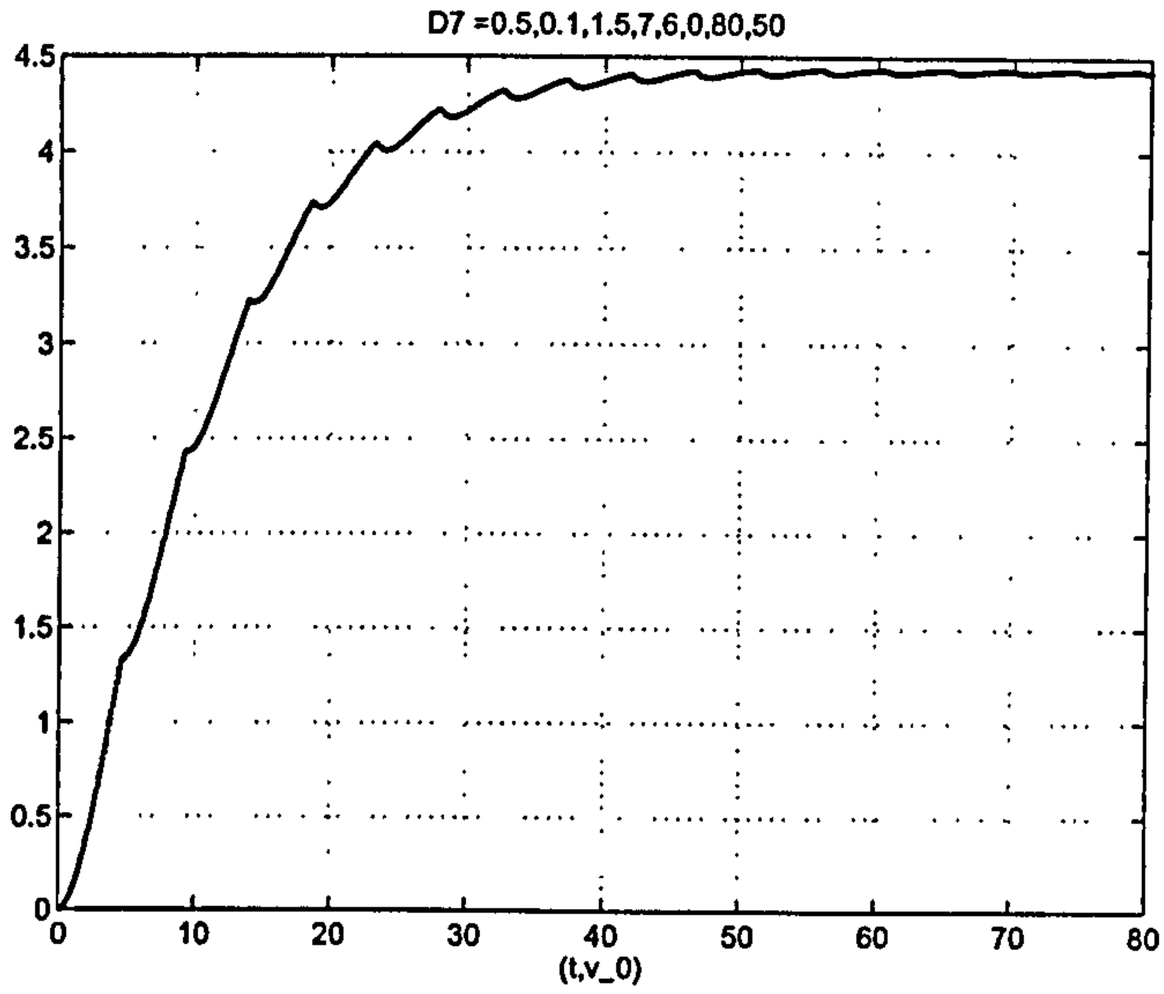


Figure 6.38: An example queue

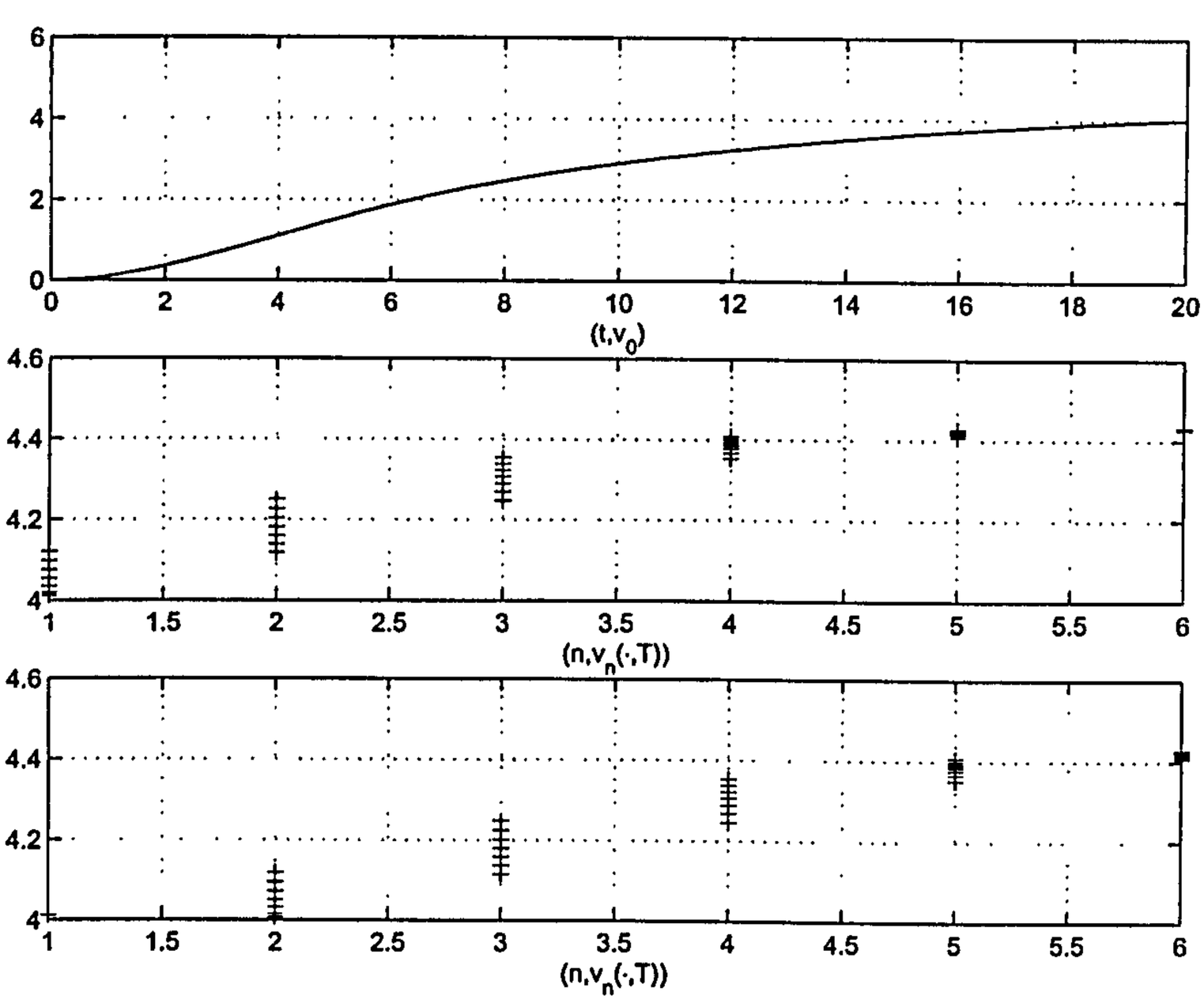


Figure 6.39: Speed of Second server is same as that of first server

Chapter 7

Towards a discrete PDP model

This chapter develops a completely new discrete renewal process which follows a deterministic motion along the positive $x-$ axis on a lattice, with occasional jumps, the latter occurring according to some discrete probability distribution. This approach provides an analysis of the relation between the constructed renewal process and the PDP Markov process. The generator of such a renewal process is derived and it is shown to satisfy a system of *delayed partial difference equations (PDEs)*. Some probabilistic characteristics of PDEs are given, including the recovery of the discrete renewal equation from the discrete generator. The discrete theory is extended to cover applications arising from queueing systems. The motivation for developing a discrete version of PDP renewal processes is indeed to provide a queueing framework in which one can analyze behaviours of computer systems. In many applications of stochastic models for computer systems, the governing equations relate discrete changes of variables, rather than rates of change. Thus, the concept of time are measured in discrete times and it has been very difficult employing continuous time Markov theory. The first step towards producing a Markov model for modeling cache coherency was taken up by (Field and Harrison 1995). There were some difficulties in formulating this problem using continuous time Markov theory. This work was restricted to steady state transition probabilities and mean value analysis.

The generator for continuous time renewal processes was derived for the first time in (Davis 1984). Davis devoted his effort to the a continuous time theory. In this previous chapters, we have adopted the general approach that connects PDPs and renewal processes. Here, we have extended it to discrete processes. We need to develop a discrete theory because such analysis are not found in the literature. In additional, some of the methods of analysis in a discrete theory will be very

different. Moreover, the collection of *recipes* necessary to extend PDP theory to discrete problems require incorporating analysis in discrete mathematics that possess *combinatorial characteristics.* The key observation about this discrete PDPs is that the stochastic "jumps " are related to some recurrent relations in the study of *partial difference equations.*

The goal of the present chapter is to present a unified description of a discrete PDP renewal through the generator of the process. The introduction of discrete partial difference equations techniques unified the entire subject, eliminates *adhoc* methods that may seem alluring. It also provides a starting point for expanding the applications of the theory to queues. More over, a discrete framework provides a vantage point from which the theory of PDP can be amenable for further meaningful analysis. The central message is the symbiosis between delay partial difference equations and renewal processes which are covered in the first few sections. We have tried to present the essential features in a reasonable palatable way.

This is how this chapter is organised. The first section (7.1) develops the essential features and displays analogies with correspondent treatments of continuous time PDP. In section(7.2) , we examine the application to renewal processes, providing the underlying discrete equations so as to complement the model as well as to illustrate its usefulness. In the rest of the chapter many of the ideas arising in discrete PDP are illustrated in a number of queues.

7.1 Discrete PDP model

Consider the problem of replacing items that encounter failure during operation, e.g. light bulbs. The initial time is set at $t = 0$ and the process obtains the first renewal at $t = Z_1$ for an item that fails. The second and subsequent renewals occur at $t = Z_1 + Z_2, Z_1 + Z_2 + Z_3, \ldots$. This formulation assumes that time t is divided into discrete units. Let f_n be the probability that a new item that is installed at a given time will fail after exactly n time units and $f_0 = 0$. Moreover, the events " a renewal occurs for the first time at time instant $t = n$ " are mutually exclusive, and are therefore

$$f = \sum_{n=1}^{\infty} f_n = 1. \tag{7.1.1}$$

Otherwise, if $f < 1$ then the density is defective with a defect of $1 - f$. We assume that $f = 1$ and introduce a random variable Z which is the waiting time with distribution

$$P\{Z = n\} = f_n.$$

Hence, a discrete renewal process is a point process $\{N_t\}$, defined

$$N_t = \sum_{i=1}^{\infty} I_{(t \geq S_i)}, \tag{7.1.2}$$

where $I_{(t \geq S_i)}$ is an indicator function, S denote the arrival times, $S_0 = 0$ and the inter–arrival times $Z_i = S_i - S_{i-1}$, i.e. $S_i = Z_1 + Z_2 + \ldots + \ldots Z_i$ for $i \geq 2$. In fact the sample path of N_t are steps functions with jumps of unit height at each S_i.

We now turn to the construction of the distribution mass function f_n. Consider n independent trials, each of which have only two possible outcomes, *Head* or *Tail.* the probability of *Head* in the k^{th} trial is p_k and that of *Tail* is $q_k = 1 - p_k$. If $p_k = p$, a constant, this construction reduces to the famous *Bernoulli trials.* Now, define

$$f_n = q_0 q_1 q_2 \ldots q_{n-2} p_{n-1}. \tag{7.1.3}$$

Suppose, we assume that the process is renewed at time $t = 0$ and let $q_0 q_1 q_2 \ldots q_{n-2} q_{n-1}$ be the probability that a renewal does not take place after time step n. From this time $t = 0$ the infinite product

$$\Pi_{n=c}^{\infty} q_n = \lim_{n \to \infty} q_0 q_1 q_2 \ldots q_{n-2} q_{n-1} q_n$$

is the probability that the system will not renews itself. This product needs to converge zero to ensure that the initial time $t = 0$ is *persistent* and so are the other times.

The above construction satisfied our restriction in (7.1.1) imposed on $\{f_n.\}$ To see this, we can arrive at this by direct computation from the probability f_n. The series

$$f = \sum_{n=1}^{\infty} f_n = 1 - q_0 + q_0(1 - q_1) + q_0 q_1 (1 - q_2) \ldots = 1 - \Pi_{n=c}^{\infty} q_n = 1,$$

in keeping with (7.1.1). Therefore, $\Pi_{n=c}^{\infty} q_n = 0$.

We are now equipped with a renewal process $\{N_t\}$ which " renews at each " renewal time " S_i. Let $u_n = E[N_n]$ be the expected number of renewals made at

time $t = n$, conditioning on $u_0 = N_0 = 0$. It is not hard to see that

$$E[N_n] = E[E\{N_n|\}] = E[0I_{T_1>n} + 1 + u_{n-S_1}I_{T_1<n}].$$

Since S_m has a mass function f_m this gives

$$u_n = \sum_{m=0}^{n} \{f_{n-m}u_m\} + b_n, \tag{7.1.4}$$

where $b_n = \sum_{m=0}^{n} f_m$. The simplest of this discrete renewal process is the *Binomial* process. In a Binomial process the " inter renewal time " is a random variable Z with $R_n = P[Z > m]$, where Z has the property: The conditional probability that the a renewal occurs at m time step, assuming that it has not occur before is a constant $1 - p$ that appears in a Bernoulli trial. This assumption leads to $f_n = p(1-p)^{n-1}$ for the probability of Z taking the values of $0, 1, 2, \ldots$, so that Z is a has a Geometric distribution. This distribution satisfies a Markov " memoryless property: the conditional distribution is just the same as the unconditional distribution of Z, regardless of the elapsed time.

Now consider the conditional probability c_n that a failure occurs at time n given that at time zero the age was k. This is given as

$$c_n = \frac{f_{n+k}}{R_k},$$

where

$$R_m = P[Z > m] = f_{m+1} + f_{m+2} + \ldots = 1 - \sum_{n=0}^{m} f_n. \tag{7.1.5}$$

If $k-1$ time units are already expended then the conditional probability of $Z = k$, is therefore $f_k/R_{k-1} = p$. The assumption behind a Binomial process is that for all k :

$$\frac{f_k}{R_{k-1}} = p, \text{ constant value}, \tag{7.1.6}$$

which satifies that lack of memory property connected with the Markovian process.

7.2 Generator of discrete renewal process

As for a general renewal process, the assumption is that f_k/R_{k-1} is no longer a constant but is a function of k. Associated with this general renewal process is the natural filtration $\Im_t = \{N_s : s \leq t\}$. Then the process $\{N_t, \Im_t\}$ is not a Markov

process. To see this complication, let $\ldots S_{i-1}, S_i, S_{i+1}, \ldots$, be the renewal times of the process with inter–renewal time distribution G. If S_i denote the last renewal time prior to some fixed time s, then the residual inter–renewal time $S_{i+1} - s$ depends on the filtration $\Im_s$ – in fact this depends on the time $\tau = s - S_i$ since the last renewal. So we can not predict the future from the past. Let $x_t = (n_t, \tau_t)$, where τ_t is the time that has expended without any renewal. In addition, consider a modified process with state $X_t = (x_t, t)$. Both processes $\{x_t\}$ and $\{X_t\}$ are Markov. Define a finite horizon cost in the time interval $t \leq l \leq T$:

$V(x) = E_x \left[\int_0^T f(x_l, l) dl + \Omega_k(\tau_T) \right]$. Here the expectation E_x is taken over the finite time interval $[t, T]$, conditioning on the state $x = (k, \tau, t)$. In control theory if $\Omega_{k_T}(\tau_T) = 0$, then the problem is said to be in *Lagrange form* and if $f(x_s) = 0$, the problem is in the *Mayer form.*

The new process X_t is now defined on the state space $(E \times [t, T]) \cup \{\Delta\}$. Δ is the usual cemetery state, defined in such that $X_t = \Delta$, for $t \geq T$. X_t is transfer immediately to Δ and stays there thereafter. Define $f(\Delta) = 0$. With these changes the expectation equation now becomes

$$V(x, t) = E_{x,t} \left\{ \sum_{s=0}^{\infty} f(x_s) \right\}, \tag{7.2.1}$$

where $f(x, T) = \Omega(.)$ for $x \in E$. Then a typical state is $(x, t) = (k, \tau, t), t \in (0, T]$.

It can be shown that $V_k(n, t)$ satisfies the system of *partial difference equations*

$$\begin{aligned} V_k(n,t) &- p_n V_{k+1}(0, t+1) + q_n V_k(n+1, t+1) + f_{k,n}(t) = 0 && (7.2.2) \\ & k = 0, 1, 2, \ldots, n = 1, 2, \ldots, t = 1, 2, \ldots, \ t \leq T && (7.2.3) \\ V_k(n,T) &= \Omega(x_T) && (7.2.4) \end{aligned}$$

where, $p_n + q_n = 1$;

7.3 Discrete Renewal Theorem

In the continuous theory (Davis 1984) it was demonstrated for the first time that the solution of the generator equation of the renewal process satisfy the renewal equation. We need to show also that the renewal equation (7.1.4) is also satisfy using the generator of the discrete renewal process. According to the partial difference equation, we set $\Omega_k(T) = k_T = k$. From its relation to the expected

number of renewals, we have a solution

$$u_k(n,s) = E_{(k,n)}\left[k_T\right]. \tag{7.3.1}$$

It is easy to see that $u_{k+k'}(n,s) = k' + U_k(n,s)$ for some integer values k and k'. Thus, $u_{k'}(0,s) = k + u_0(0,s)$. From the generator of the renewal process in the time interval $[0,m]$, where $T = m$:

$$\begin{array}{rcrcl}
V_0(m-1,m-1) & = & p_{(m-1)}V_1(0,m) & + & q_{(m-1)}V_0(m,m) \\
V_0(m-2,m-2) & = & p_{(m-2)}V_1(0,m-1) & + & q_{(m-2)}V_0(m-2,m-1) \\
V_0(m-3,m-3) & = & p_{(m-3)}V_1(0,m-2) & + & q_{(m-3)}V_0(m-2,m-2) \\
V_0(m-4,m-4) & = & p_{(m-4)}V_1(0,m-3) & + & q_{(m-4)}V_0(m-3,m-3) \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
V_0(1,1) & = & p_1V_1(0,2) & + & q_1V_0(2,2) \\
V_0(0,0) & = & p_0V_1(0,1) & + & q_0V_0(1,1),
\end{array}$$

with boundary condition $V_0(m,m) = 0$. Also,

$$\begin{array}{rcl}
V_1(0,m-1) & = & 1 + V_0(0,m-1), \\
V_1(0,m-2) & = & 1 + V_0(0,m-2), \\
V_1(0,m-3) & = & 1 + V_0(0,m-3), \\
\cdots & \cdots &
\end{array}$$

Using method of recursive substitution, the system reduces to

$$
\begin{aligned}
V_0(m-1,m-1) &= p_{(m-1)}V_0(0,m) + q_{(m-1)}V_0(m,m) + p_{(m-1)},\\
V_0(m-2,m-2) &= p_{(m-2)}V_0(0,m-1) + q_{(m-2)}p_{(m-1)}V_0(0,m)\\
&+ q_{(m-2)}q_{(m-1)}V_0(m,m) + q_{(m-2)}p_{(m-1)} + p_{(m-2)},\\
V_0(m-3,m-3) &= p_{(m-3)}V_0(0,m-2) + q_{(m-2)}p_{(m-2)}V_0(0,m-1) + q_{(m-2)}p_{(m-1)}V_0(0,m)\\
&+ q_{(m-3)}q_{(m-2)}q_{(m-1)}V_0(m,m)\\
&+ q_{(m-3)}q_{(m-2)}p_{(m-1)} + q_{(m-3)}p_{(m-2)} + p_{(m-3)}\\
V_0(m-4,m-4) &= p_{(m-4)}V_0(0,m-3) + q_{(m-4)}p_{(m-3)}V_0(0,m-2)\\
&+ q_{(m-4)}q_{(m-2)}p_{(m-2)}V_0(0,m-1) + q_{(m-4)}q_{(m-2)}p_{(m-1)}V_0(0,m)\\
&+ q_{(m-4)}q_{(m-3)}q_{(m-2)}q_{(m-1)}V_0(m,m)\\
&+ q_{(m-4)}q_{(m-3)}q_{(m-2)}p_{(m-1)}\\
&+ p_{(m-4)}q_{(m-3)}p_{(m-2)} + q_{(m-4)}P_{(m-3)} + p_{(m-4)}\\
V_0(m-5,m-5) &= p_{(m-5)}V_0(0,m-4) + q_{(m-5)}p_{(m-4)}V_0(0,m-3)\\
&+ q_{(m-5)}q_{(m-4)}p_{(m-3)}V_0(0,m-2)\\
&+ q_{(m-5)}q_{(m-4)}q_{(m-2)}p_{(m-2)}V_0(0,m-1) + q_{(m-5)}q_{(m-4)}q_{(m-2)}p_{(m-1)}V_0(0,m)\\
&+ q_{(m-5)}q_{(m-4)}q_{(m-3)}q_{(m-2)}q_{(m-1)}V_0(m,m)\\
&+ q_{(m-5)}q_{(m-4)}q_{(m-3)}q_{(m-2)}p_{(m-1)}\\
&+ q_{(m-5)}p_{(m-4)}q_{(m-3)}p_{(m-2)}\\
&+ q_{(m-5)}q_{(m-4)}p_{(m-3)} + q_{(m-5)}p_{(m-4)} + p_{(m-5)}
\end{aligned}
$$

Therefore this recurrent relation will continue up to origin at $(0,0,0)$ with the expected number of renewals $V_0(0,0)$. This is the total number of renewals that have occurred in the time interval $[0,m]$. Observe the following notations

$$
\begin{aligned}
u_0 &= V_0(0,m) = 0,\\
u_1 &= V_0(0,m-1),\\
u_2 &= V_0(0,m-2),\\
\cdots &\quad \cdots\\
u_{m-1} &= V_0(0,1)\\
u_m &= V_0(0,0).
\end{aligned}
$$

If $m = 5$ then

$$\begin{aligned}
f_0 &= p_{(m-5)} = p_0 \\
f_1 &= q_{(m-5)}p_{(m-4)} = q_0p_1 \\
f_2 &= q_{(m-5)}q_{(m-4)}p_{(m-3)} = q_0q_1p_2 \\
f_3 &= q_{(m-5)}q_{(m-4)}q_{(m-2)}p_{(m-2)} = q_0q_1q_2p_3 \\
f_4 &= q_{(m-5)}q_{(m-4)}q_{(m-2)}p_{(m-1)} = q_0q_1q_2q_3p_4 \\
f_5 &= q_{(m-5)}q_{(m-4)}q_{(m-2)}p_{(m-1)} = q_0q_1q_2q_3q_4p_5
\end{aligned}$$

In general this recurrent relation has established the following.
$V_0(m,m) = V_0(0,m) = 0$. The renewal process is defined on the time $[0,m]$.
$u_m = V_0(0,0)$ is the expected number of arrivals up to time m. We have seen above that u_m is decomposed into a linear combinations of a number of terms. In general, it says that if the first arrival occurs at time s then the expected number of arrivals in the remaining time up to m is u_{m-s}. We see from the above construction that event ϵ_s " *an arrival occur at time so that the expected number of replacements in the remaining time is* u_{m-s} ". The event ϵ_s is equivalent to $1 + u_{m-s}$ which occurs with probability f_s, and summing across the various possible values of s gives

$$u_m = \sum_{s=0}^{m} \{f_{m-s}(1+u_s)\} = \sum_{s=0}^{m} \{f_{m-s}u_s + f_s\}. \tag{7.3.2}$$

This is what we set out to show.

7.4 Discrete approximation of Renewal Density

In the continuous time PDP theory we are usually provided with the Renewal time distribution $G(\tau)$ with density $g(\tau)$ and harzard rate $\mu(\tau)$. Let τ be the time that has elapsed since the last renewal. Thus, the probability that a renewal will occur in $(\tau, \tau + \Delta_x)$ is

$$g(\tau) = \mu(\tau)exp\left[-\int_0^{\tau} \mu(x)dx,\right] \tag{7.4.1}$$

where $\mu(\tau)\Delta_x$ is the probability that a system which survives to τ will be renewed in $(\tau, \tau + \Delta_x)$.

$$\overline{G} = P[X > \tau] = \int_{\tau}^{\infty} g(s)ds. \tag{7.4.2}$$

Now define the continuous harzard rate $\mu(\tau)$ as

$$\mu(\tau) = \frac{g(\tau)}{1 - G(\tau)} = \frac{\dot{G}(\tau)}{1 - G(\tau)}. \tag{7.4.3}$$

Now consider

$$exp\left(-\int_t^s \mu(\tau)d\tau\right) = exp\left(\int_t^s \frac{d}{d\tau}(\log(1 - G(\tau)\right) = \frac{1 - G(s)}{1 - G(t)}$$

Thus, the integral equation relating the hazard rate and the survival probability is

$$\frac{d}{d\tau}\overline{G} - \mu(\tau)\overline{G} = 0, \ \ \overline{G}(0) = 1. \tag{7.4.4}$$

For a **Gamma distribution** the density $g(\tau)$ is given as

$$\frac{\lambda^{\alpha}\tau^{\alpha-1}}{\Gamma(\alpha)}e^{-\lambda\tau}$$

where Γ is the complete gamma function defined by

$$\Gamma(\alpha) = \int_0^{\infty} e^{-t}t^{\alpha-1}dt.$$

The distribution $G(\tau)$ is defined as

$$G(\tau) = \frac{1}{\Gamma(\alpha)}\int_0^{\lambda\tau} e^{-t}t^{\alpha-1}dt.$$

If the shape parameter α is positive integer k, the Gamma distribution is just a well–known Erlang(k) distribution for which

$$\frac{\lambda^{k}\tau^{k-1}}{(k-1)!}e^{-\lambda\tau}$$

and

$$G(\tau) = 1 - \sum_{j=0}^{k-1} e^{-\lambda\tau}\frac{(\lambda\tau)^j}{j!}.$$

This distribution is simply a sum of k independent exponential distribution with a common rate λ. Therefore, the harzard rate can be obtained by evaluating the ratio $g(\tau)/(1 - G(\tau)$. For a **Weibull dsitibution** the harzard rate takes a simpler form $\alpha\lambda(\lambda\tau)^{\alpha-1}$.

Again assume that the renewal process has a continuous distribution function $G(.)$

and density g. Define a function

$$Z(X) = m, \text{ if } m \leq X < m+1,$$

where m is any integer. Then $Z(X)$ is an integer valued discrete random variable with mass function

$$q_m = P[m \leq Z < m+1] = \int_m^{m+1} g(w)dw.$$

Obviously, $P[Z(X) \leq Z] = 1$, $\sum_{m=0}^{\infty} q_m = 1$, $G_Z(x) \leq G(x)$ and $\|Z(X) - X\| \leq 1$, where $G_Z(x)$ is a discrete approximation of G.

We now turn to how to construct q_n then f_n. We have seen in (7.1.5) that in a discrete case the survival distribution is

$$R_k = P[\tau > k] = P[Z > k] = f_{k+1} + f_{k+2} + \ldots = 1 - \sum_{n=0}^{k} f_n.$$

Now $f_k = R_{k-1} - R_k$, and hence

$$\frac{f_k}{R_{k-1}} = q_k = \int_m^{m+1} g(w)dw$$

For fixed integer n, with $0 \leq n \leq r \leq 2^n - 1$, and $k \geq 1$, define

$$Z_n(X) = k + r2^{-n}, \text{ if } = k + r2^{-n} \leq X <= k + (r+1)2^{-n}.$$

Then $Z_n(X)$ is an integer valued discrete random variable taking values in $(k + r2^{-n} : 0 \leq n \leq r \leq 2^{-n} - 1, k \geq 0)$ with mass function

$$q_m = P\left[Z_n(X) = k + r2^{-}\right] = \int_{k+r2^{-n}}^{k+(r+1)2^{-n}} g(w)dw.$$

Again we have $Z_n(X) \leq X$, but this time, by the approximation,

$$|Z_n(X) - X| \leq 2^{-n}.$$

Thus, by choosing n large enough, we can find a discrete random variable $Z_n(X)$ such that $|Z_n(X) - X|$ is as small as possible.

In the continuous theory an Erlang(k) distribution plays a major role because quite a large class of distributions can be approximated with it. In the discrete case the

Pascal distribution is the equivalent of the Erlang(k) distribution and is given as

$$f_m = P[Z = m] = \begin{pmatrix} m+k-1) \\ k \end{pmatrix} p^k (1-p)^m.$$

To appreciate this distribution, consider a succession of Bernoulli trials and let Z be the *waiting time* untill the k^{th} success to turn up. It is necessary that the number of trials be $k+m$ (where $m = 0, 1, 2,$) and we inquire the probability that the k^{th} success at the trial number $k+m$. This is just f_m which reduces to the Geometric distribution for $k = 1$.

To construct the generator of a renewal process with a Pascal density we need a discrete harzard rate functions $\{q_m\}$ that expands

$$\begin{pmatrix} m+k-1) \\ k \end{pmatrix} p^k (1-p)^m = q_0 q_1 q_2 \cdots q_{m-2} p_{m-1}.$$

This is non–trivial and we shall return to investigate. Instead, we consider the following state (n, r, t) denote the state of the time t, where there n renewals in the systems with the understanding that r is number of successes so far in the Pascal–Bernoulli set up and there are a total of k successes for a renewal to occur. Here, r runs from 1 to k.

$$\begin{aligned} V_n(r,t) &= pV_k(r+1,t+1) + qV_k(r,t+1) + f_{n,r}(t). && (7.4.5) \\ & r = 0,1,2,\ldots,k, k = 1,2,\ldots,t = 1,2,\ldots, \quad t \leq T && (7.4.6) \\ V_n(k,t) &= V_{n+1}(0,t). && (7.4.7) \end{aligned}$$

7.5 Discrete model of $M/D/1/N$ queue

A continuous time PDP solution of the $M/D/1N$ queue problem was given in (Davis and Howl 1997). Now the Suppose that k_t represents the number of jobs in the $M/D/1N$ queue at discrete time t. Associated with this is the natural filtration $\Im_t = \{n_s : s \leq t\}$. Then the process $\{n_t, \Im_t\}$ is not a Markov process. This is because, if S_k denotes the last service time prior to some fixed time t, then the residual inter–service time $T_{k+1} - t$ depends on the filtration $\Im_t$– in fact this depends on the time $\tau = t - S_k$ since the last service. We have a constant–service distribution

$$B(t) = \begin{cases} 0 & \text{if} \quad 0 \leq t < D \\ 1 & \quad\quad D \leq t < \infty. \end{cases}$$

Using the framework of PDP one can conceive the $M/D/1N$ as a two–component process $x_t = (k_t, \tau_t)$, where τ_t is the time since the last jump. Now $\{x_t\}$ is a Markov process and it has a sample path similar to that provided in this example. Assume that the process may start at state $(2,0)$*and* that no job arrives in the time interval greater than D. The state of the system only changes in steps Δ_x on the x–axis

$$(k,0) \rightarrow (k,1) \rightarrow (k,2) \rightarrow \ldots$$

until it hits the boundary point D and moves to $(0,0)$.

The process then moves again along the x–axis with state $(1,n) : 0 \leq n \leq D$. As soon as a new job arrives the state jumps to $(2,n)$. Therefore, the generator of this process is the set of system equations:

$$\begin{aligned}
V_N(n+1,t+1) &- V_N(n,t) + f_{N,n}(t) = 0, && (7.5.1)\\
pV_{k+1}(n+1,t+1) &+ qV_k(n+1,t+1) && (7.5.2)\\
&- V_k(n,t) + f_{k,n}(t) = 0 && (7.5.3)\\
k = 2,3,\ldots,N, &\quad n = 1,2,\ldots,t = 1,2,\ldots,\ t \leq T && (7.5.4)\\
pV_2(n+1,t+1) &+ qV_1(n+1,t+1) && (7.5.5)\\
&- V_1(n,t) + f_{1,n}(t) = 0 && (7.5.6)\\
pV_1(0,t+1) &+ qV_0(0,t+1) && (7.5.7)\\
&- V_0(0,t) + f_{1,0}(t) = 0 && (7.5.8)\\
\text{with boundary conditions:} & && (7.5.9)\\
V_k(n,T) &= g_{k,n}, k = 1,1,2,\ldots,N, n \in [0,D] && (7.5.10)\\
V_0(0,T) &= g_0, && (7.5.11)\\
V_1(D,t) &= V_0(0,t) = V_0(t), && (7.5.12)\\
V_k(D,t) &= V_{k-1}(0,t), k = 1,1,2,\ldots,N-1,N, t \in [0,T] && (7.5.13)
\end{aligned}$$

where $g(x_T)$ and $f_{k,n}(t)$ are given functions. Then

$$V_k(n,t) = E_{x,t}\left[\sum_{s=t}^{T} f_{k,n}(s) + g(x_T)\right] \qquad (7.5.14)$$

is the solution of the system of partial difference equations (7.5.1)–(7.5.7).

An important performance parameter in queue system is the expected number of

jobs in the queue at time. This is denoted as $E[q_T|k_0 = 0]$. To evaluate this, we set

$$g_{k,n} = k-1, k = 1, 1, 2, \ldots, N, n \in [0, D], \quad (7.5.15)$$

$$g_0 = 0, \quad (7.5.16)$$

$$f_{k,n} = 0, k = 1, 1, 2, \ldots, N, n \in [0, D], \quad (7.5.17)$$

$$f_0 = 0. \quad (7.5.18)$$

Using (7.5.14), we can obtain a solution of the expected queue length at any time in $[0, T]$.

It is equally important to observe that when the buffer is full dynamic of the system is described by the partial difference equation:

$$V_N(n,t) - V_N(n+1, t+1) = 0, \quad (7.5.19)$$

which is a discrete hyperbolic partial difference equation. To see that it is hyperbolic, the equation can be re–arranged in the form

$$\frac{\{V_N(n,t) - V_N(n, t+\Delta_t)\}}{\Delta_t} - \frac{\{V_N(n+\Delta_x, t+\Delta_x) - V_N(n, t+\Delta_t)\}}{\Delta_x} = 0, \quad (7.5.20)$$

In the limit as $\Delta_x \to 0$ and $\Delta_t \to 0$, we have

$$\frac{\partial}{\partial t} V_N(x,t) + \frac{\partial}{\partial x} V_N(x,t) = 0,$$

which agrees with the continuous case (Davis and Howl 1997).

7.5.1 Discrete model for $M/G/1/N$ queue

The M means a Markov, i.e. the inter–arrival times distribution is geometric with a constant parameter p. The service process is given a distribution:

$$P[\tau = n] = q_0 q_1 q_2 \ldots q_{n-2}(1 - q_{n-1}), \ 0 < q_n < 1, \ n = 1, 2, \ldots, \quad (7.5.21)$$

$$P[\tau \geq D] = 0, \ n = 1, 2, \ldots. \quad (7.5.22)$$

(7.5.23)

We truncate the service time at some large D where $q_0 q_1 q_2 \ldots q_{D-2}(1 - q_{D-1}) \approx 0$.

Define

$$r_n = 1 - p - q_n, n = 1, 2, \ldots, D.$$

Then, if a jump occurs it may be because a job increment has arrived, because a job has been completed, these events having probabilities p and q_n, respectively. On the other hand if $n = D$ then a service must complete since D is the maximum service time. Therefore, the transition measure $Q(., x = (k, n))$ following a jump at time t from a position x_t- can be computed. Thus, the generator of this process is the set of system equations:

$$V_N(n+1,t+1) - V_N(n,t) + f_{N,n}(t) = 0, \tag{7.5.24}$$

$$pV_{k+1}(n+1,t+1) + q_nV_{k-}(0,t+1) - r_nV_k(n+1,t+1 - V_k(n,t) + f_{k,n}(t) = 0 \tag{7.5.25}$$

$$k = 2,3,\ldots,N, n = 1,2,\ldots,t = 1,2,\ldots, \quad t \leq T \tag{7.5.26}$$

$$pV_2(n+1,t+1) + q_nV_1(0,t+1) - r_nV_1(n+1,t+1) - V_1(n,t) + f_{2,n}(t) = 0 \tag{7.5.27}$$

$$pV_1(0,t+1) + pV_0(0,t+1) - V_0(0,t+1) - V_0(0,t) + f_{k,0}(t) = 0 \tag{7.5.28}$$

with boundary conditions: (7.5.29)

$$V_k(n,T) = g_{k,n}, k = 1,1,2,\ldots,N, n \in [0,D] \tag{7.5.30}$$

$$V_0(0,T) = g_0, \tag{7.5.31}$$

$$V_1(D,t) = V_0(0,t) = V_0(t), \tag{7.5.32}$$

$$V_k(D,t) = V_{k-1}(0,t), k = 1,1,2,\ldots,N-1,N, t \in [0,T] \tag{7.5.33}$$

7.6 Markov chain models for queues

7.6.1 $M/D/1/N$

$M/D/1/N$ has Binomial arrival and deterministic service time. Define the following:

- N = Buffer size,
- k = number of jobs in the system not including the job in system,
- τ = elapsed service time,
- D = is the service time for the deterministic case,
- s = is the time left before job completion, $s = D - x$,
- e = the amount of work in system for the server to do, $e = kD + s$.

Expected value function $V_e(t)$ in the finite interval $[t, T]$:

$$V_{kD+s}(t) = pV_{(k+1)D+s-1}(t+1) + qV_{kD+s-1}(t+1), \tag{7.6.1}$$

boundary conditions

- $V_{kD+s}(T) = k - 1$, for queue length computation,
- $V_{kD+s}(T) = 1$, for $k = N - 1$ and zero otherwise, for blocking probability $P_N(t)$,
- $V_{kD+s}(T) = 1$, for $k = 0$, and zero otherwise, for server idle probability $P_0(t)$,
- $V_{kD+0}(t) = V_{(k-1)D+D}$.

7.6.2 $M/E_L/1/N$

$M/E_L/1/N$ has a Poisson arrival and Erlang service with L exponential service phases.

- N = Buffer size
- k = number of jobs in the system not including the job on the server,
- L = is the number of phases in server , each phase having an exponential (μ) holding time,
- x = no of stages completed so far,
- s = is the time left before job completion = no. of phases left for job completion $s = L - x$,
- e = the amount of work in system for the server to do, $e = kL + s$.

Each arrival increases the number of phases in the system by L, and each time a phase is completed, the total number of phases in the system decreases by unity
Expected value function $V_e(t)$ in the finite interval (t, T) :

$$V_{kL+s}(t) = \lambda V_{(k+1)L+s}(t+1) + \mu V_{kL+s-1}(t+1) + (1 - \lambda - \mu)V_{kL+s}(t+1) \tag{7.6.2}$$

with boundary conditions

- $V_{kL+s}(T) = k - 1$, for queue length computation,
- $V_{kL+s}(T) = 1$, for $k = N - 1$ and zero otherwise, for blocking probability $P_N(t)$,
- $V_{kL+s}(T) = 1$, for $k = 0$, and zero otherwise, for server idle probability $P_0(t)$,
- $V_{kL+0}(t) = V_{(k-1)L+L}$.

In terms of probability, let P_{kL+s} be the probability that there are k jobs in the system and the job currently in service has s phases left to be completed. Each arrival increases the total number of phases by L, and each time a phase is completed, the number of phases in the system decreases by unity. Thus,

$$P_{kL+s}(t+\Delta_t) = \lambda\Delta_t P_{(k-1)L+s}(t)+\mu\Delta_t P_{kL+s+1}(t)+[1-(\lambda+\mu)\Delta_t]\,P_{kL+s}(t) \quad (7.6.3)$$

for $(k \geq 0, s > 0)$.

$$P_{0L+0}(t+\Delta_t) = (1-\lambda\Delta_t)P_0(t) + \mu\Delta_t P_{0L+1}(t). \quad (7.6.4)$$

for $(k = 0, s = 0)$. with boundary conditions

- $P_j = 0$ for $(k = 0, s < 0)$.
- $P_{kL+s}(0) = 1$, for $k = N - 1$ and zero otherwise, for blocking probability $P_N(t)$.

These equations state that the probability $P_n(t+\Delta_t)$ of n phases in the system at time $(t+\Delta_t)$ equals to the probability $P_n(t)$ of n phases in the system at time t multiplied by the probability of no arrivals and no departures, plus the probability $P_{n-1}(t)$ of $n-1$ phases in the system at time t multiplied by the probability of 1 arrival and no departures, plus the probability $P_{n+1}(t)$ of $n+1$ phases in the system at time t multiplied by the probability of no arrival and a single departure. Note that the probability of no arrivals and no departures is $(1-\lambda\Delta_t)(1-\mu\Delta_t)$. The term involving $(\Delta_t)^2$ falls out in forming the difference equations.

7.7 Conclusion

This presentation is an outgrowth of recent research into the application of PDP models to the development of computing servers. This work deals with variables

that are related through discrete equations. The results provide a brief integrated development of a discrete PDP process and the applications of it to renewal and queueing systems. Serious attempt have been made to validate the models[1] and to find and prove discrete analogues of existing continuous theorems. In particular, a discrete renewal theorem has been proved for the first time using the PDP model. This extends to a new treatment of transient queueing problems which permits formulation in a discrete environment.

It is hope that the result may be useful as a reasonable self-contained rudimentary model and also as a stepping stone toward the less elementary treatments.

[1]Private communication with Davis.

Chapter 8

Data communication across an IBM switch

8.1 Introduction

In this chapter, we show how message passing between two server nodes connected on an IBM switch can be modelled using a transient PDP queue model. This is an $M/G/1/N$ queue that is modelled with a Markov theory, a class of Markov processes known as piecewise–deterministic Markov processes (PDPs). The PDP model provides methods for evaluating and optimising the performance of data communication system for message passing. There is a particular emphasis in the optimisation of bandwidth through the I/O hardware adaptor and maximising the utilisation of this bandwidth by the underlying software protocol. All the necessary analysis tools have been developed for the first time and applied to a real scientific benchmark running a parallel MPI application.

The case study in data communication in this chapter represents a significant improvement in the approach of PDP research. The improvement is actually a major leap forward from abstract mathematics that have characterised PDP research to aspects that enter into genuine system design. The central task is to find the basic elements of the theory that offer solutions to network problems. As a way forward, one must, in the first place, show that the models provided by the theory of PDP are adequate to the network traffic task in hand–that they are rich enough to account for the attested variety of media (both hardware and software) through which the message passing traffic flow and, indeed, for their possible probabilistic characteristics. Unless these tasks are met by the theory of PDP, it

will be impossible to account for the fact that the theory is rich enough for data communication problems.

As discussed in earlier chapters, the basic problem is that the scope of stochastic processes emendable to PDP analysis are quite large, including almost all well–known non–diffussion processes, whereas practical examples available are much too impoverished to provide any good hints. There is good evidence, supported by many researchers, that it is due to the complexity of the equations provided by the theory that discourages its use. As noted, there is a tension between theory and genuine application. To achieve the later, we restrict our investigation to a real IBM switch under development. This approach enables PDP concepts to be tested against available data from both software and hardware simulations.

The focus resulted in the application of the theory to data communication among servers connected on an IBM switch. This is a packet switched network with message passing functions. The switch allows individual servers to communicate with one another across high speed digital links(I/O busses and adaptors), running a parallel processing application.

The two basic questions that arise are:

1. How can the bandwidth analysis of a handware communication link lead to PDP as a queueing model?

2. Can PDP also be used to approach the problem of evaluating and optimising the software protocol for a high utilisation of that link bandwidth?

The first part of this chapter shows how the application to low level data communication through the I/O bus lead to a PDP queue model, and how the main objective of optimising data transmission is achieved. There is an emphasis on how congestion across various traffic paths affect the service distribution and the time dependent queue length at the bus. This is a major justification for using transient $M/G/1/N$ queue. The second part of the chapter is devoted to a PDP model for a high level abstraction of data communication at the software level, with an aim of estimating the time dependent amount of data contained in the software data buffer during this message passing process. This provides a general framework for evaluating system performance at a high vantage point.

In addition, the major merit of applying PDP to software protocol is that it identifies the relationship between the queue length and the bandwidth u. The

difficulty inherent in the software modelling, on the other hand, is that it aims at the maximisation of the utilisation of the link bandwidth. And this depends on the protocol processing latency and "handshaking " types of operation before and after data transmission. This protocol latency directly affects the amount of packets that are in the protocol buffer at any given time waiting to be transferred. If the protocol latency is low, the average queue length of packet is small and the utilisation of the link is high, meaning that the high proportion of the bandwidth u is used.

Furthermore, there is no general way of determining u in an accurate form. It has to be estimated in a case–by–case basis. It is affected by the kind of protocol, among other things. In this work, we examine the estimation of u under a DMA transfer operation, which is one link bandwidth problem that is amenable to a solution by direct application of PDP model to the queueing buffers at I/O port of the server node.

Reastic application of PDP to system design is missing. This chapter is an attempt to start the process of filling this gap. Therefore, our main objective here is to give an account of the main results of the application of PDP to data communication. The aim is to give some arguments in the formulation of the modelling problem that leads to the major results rather than obtaining a general results.

In a parallel processing application using the programming language MPI, a user task initiates a request on Node A to submit a message to Node B. Because the to–be transferred data is stored in user (virtual address) space, and the pages are not to be paged out by the operating system during the data transfer, the communication protocol pins(locks) down the memory area and copies the data from the user buffers into the pinned buffer area. At the next step, the protocol informs the adapter to transfer the data by initiating a direct memory access (DMA) operation referencing the pinned buffer area. A similar (but reverse) process is used on the receiver. Since MPI I/O requests so often involve data blocks of 2024($2K$) Bytes and sometimes thousands of them(measured in Mega Bytes), a DMA operation is done by a dedicated adaptor that is attached to the I/O port of the server node. This DMA operation allows the transfer of data without the intervention from the CPU. Therefore, the software communication system(an extension of the operational system running on each server node) is responsible for (1) setting up the data control blocks containing the information about the transfer (2) flow control that keeps track of the amount of packets of data that have been transferred and those that are left in the data queue. The hardware adaptor then takes data from the queue, passes it through the I/O bus

and informs the protocol when data transfer operation is completed.

It is assumed that the sender node could be running one operational system like Linux and the receiver node could be running a different operational system like IBM unix(AIX). Each node also has 8 adaptors that are attached to the I/O ports. We shall sometimes call the I/O port a GX bus and its controller(GXC). Under this scheme one feature that complicates the data communication is the fact that the hardware adaptor on the sender side does not know the physically address(PA) of the data. It therefore needs to translate the virtual address(VA) to PA before it can start reading the data. The VA is provided to the adaptor by the software communication protocol.

A node in terms of the IBM SP is defined as a single, stand–alone machine, self–contained in a "box" which is mounted in an SP frame slot. A node typically possesses the following characteristics: it own I/O ports, network adapters, independent memory resources with memory cards and caches and copy of the AIX operating system.

The IBM SP High Performance InfiniBand switch under development provides the message passing fabric that connects all of the SP processors together into a single system to run parallel applications. SP switch networks are bi–directional multistage interconnection networks, allowing internode connection for all processors to send messages simultaneously.

8.2 Motivation

8.2.1 Parallel MPI application

ASCI (Accelerated Strategic Computing Initiative) is a Department of Energy initiative to use computer simulation rather than nuclear testing to ensure the safety, reliability, and performance of the nuclear stockpile.

We used a NAS benchmark data that has been developed purposely for performance evaluation of highly parallel super computers to generate the data communication traffic. These benchmark consists of five parallel codes. Together, they mimic the computation and data movement characteristics of large scale computational fluid dynamics (CFD) applications. The NAS parallel benchmark has been developed at NASA Ames Research Center to study the performance of

parallel super computers. MPI is a library of functions which provide a standard interface for portable parallel programming. The programming paradigm in MPI is based upon message passing as the communication method between processes on different nodes.

8.2.2 Applications to parallel file systems

We pointed out in the introduction to this thesis that in recent years there have been considerable interest in client and server model of communication. Parallel file system makes it possible for several clients application to share a common server for data storage and access. The IBM has developed a General Parallel File System (GPFS) as a scalable cluster file system for the RS/6000 SP. It was originally developed for large–scale multimedia on LAN and ATM switches.

Later, it was extended to support the additional requirements of parallel computing. GPFS supports file systems of several terabytes and run at high I/O bandwidths.

On an IBM switch network, clients makes requests to transfer data to GPFS. GPFS allows parallel software codes running on as many as 512 server nodes to read and write individual data files across the switch to each other at very high bandwidth. This high I/O bandwidth rate and storage capacity is required to provide data to the simulation codes and to allow output to be stored at the rate required by the machine's tremendous computational power.

The software network protocol will inevitably fail to achieve high bandwidth utilisation through the I/O port in such massive operations because it copies data from the user buffer to the pinned buffer, a relatively expensive process with respect to bandwidth. These traditional unix(AIX and Linux) I/O interfaces are based on copy semantics, where software transfer data between the kernel and user–defined buffers. Although simple, copying data from one memory region to the other limits the ability of the operating system to efficiently implement data transfer operations. A zero copy communication system eliminates this extra copying operation.

We are currently developing a new version of zero copy protocol. One good example occurs when several client applications running on the client/server model want to read or write data in the server across a switch. Each client wants to read data from files on a server side. For example, a client may issue sequential

$2K$Bytes read operations to GPFS who finds that the data is not available on the local client memory system. This forces the GPFS file system to send remote message passing request across the switch to the server side. When the requested data blocks return to GPFS running on the client side it forwards the blocks to the application which requested them. Normally, GPFS uses a TCP/IP transport protocol for data communication. Unfortunately, TCP/IP layer requires that GPFS copies data from its data to its address space before it can sends it to the adaptor. Zero copy protocol will replace the TCP/IP protocol copy operation and makes it possible for the adaptor to transfer the data directly from the address space of GPFS.

8.3 Major achievements in data communication

We regard the result of this investigation as a major achievement for applied PDP. As we have argued repeatedly, PDP theory has a strong impetus since its creation by Davis in 1984, but it was " buried " under the burden of abstract mathematics. The computation models developed here are attempts to fill that gap between theory and practice.

This case study led to the following major achievements in the development of data communication clustering on the most powerful server (called Regatta) developed by IBM:

1. **Estimation of buffering size N :** the I/O bus on this server needed a certain optimal number of buffers for Read and Write data and a certain intermediate buffers in the Fabric bus system to sustain design bandwidth. We have found out that the sizes of these buffers affect bandwidth. Using PDP theory, we are able to formulate the I/O bandwidth optimisation problem. As a results, optimal sizes of the buffers are determined.

2. **Optimisation of link bandwidth u :** For a fixed size of buffering subsystem, the mean inter arrival time of DMA data that will optimise the I/O bandwidth is needed. An adaptor issues DMA request to the I/O at a certain rate. A large value of mean inter arrival time leads to congestion; a very small value of mean inter arrival time results in a small value of bandwidth. PDP queue model helps to optimisation the arrival rate of DMA traffic. Also note that the arrival rate here is the Poisson component in the queue model $M/G/1/N$, denoted here by M. That mean inter arrival time is

represented here by the reciprocal($1/\lambda$) of the Poisson rate λ. Using the optimal buffer size N that is estimated, the size of the DMA data(128 Bytes) and expected queue length of DMA data in the buffering system, we are to determine the value of $1/\lambda$ that makes the *blocking probability*(probability of congestion) of traffic minimum.

3. **Flow control:** Certain recommendations about the patterns of issuing DMA data that makes use of flow control algorithm for smooth traffic flow into the memory sub system were provided to adaptor designers. This was only possible because of good predictive power of PDP.

4. **PDP aid queue simulation:** In addition, we advised the queue modelling and hardware server simulation teams about the proper procedures for studying congestion across data communication paths that helped system development. Such hardware server simulations include the use of I/O sources to study the effects of DMA traffic patterns on the server system. The data record on the arrival times of DMA data and the busy (or life) times of buffers were examined. These combination of using PDP model and simulations provided us with good data for statistical analysis. Finally, we used statistical analysis methods developed in the previous chapters to validate the bandwidth results predicted from the model.

5. **Estimation of packet queue length:** The objective of predicting the transient queue length of packets in the software send queue was attained and the results were used to advise protocol developers. PDP model appears to provide our designer with a promising approach for simulating design parameters. Our experience shows that the flexibility of the framework is extremely useful for supporting a wide range of protocols, including zero copy.

8.4 Review of I/O communication models

The challenge faced with today's high-speed cluster interconnects is how to deliver the hardware performance into user address space. Traditional communication sub systems such as TCP/IP are not capable of approaching the capacity of the hardware. The argument made in this study is that only a software protocol combined with an efficient I/O sub system design has the capability to provide high bandwidth link. As outlined and discussed by (Araki 1998a, Bhoedjang 1998a, Tezuka 1998, Warschko 1998a, Dubnicki 1998) several software

communication subsystems have been proposed and implemented over the last few years. The basic goal of all these sub systems is to minimise the involvement operating system component of communication, allowing the software protocol direct use of a high hardware I/O bandwidth, and to avoid expensive copy as well as buffering operations along the software layer. In figure(8.1) , Bullet 1 represents a DMA transfer initiated by the cluster interconnect. Bullet 2 depicts a Programmed I/O (PIO) control sequence. Bullet 3 shows a PIO data transfer between the sender and the cluster interconnect. Bullet 4 outlines another PIO data transfer between the cluster interconnect and the receiver. Bullet 5 represents a cluster interconnect initiated DMA control sequence transfer, whereas Bullet 6 depicts an interconnect initiated DMA data transfer. Bullet 7 and 8 show memory copy operations between kernel and user address space. The Programmed I/O

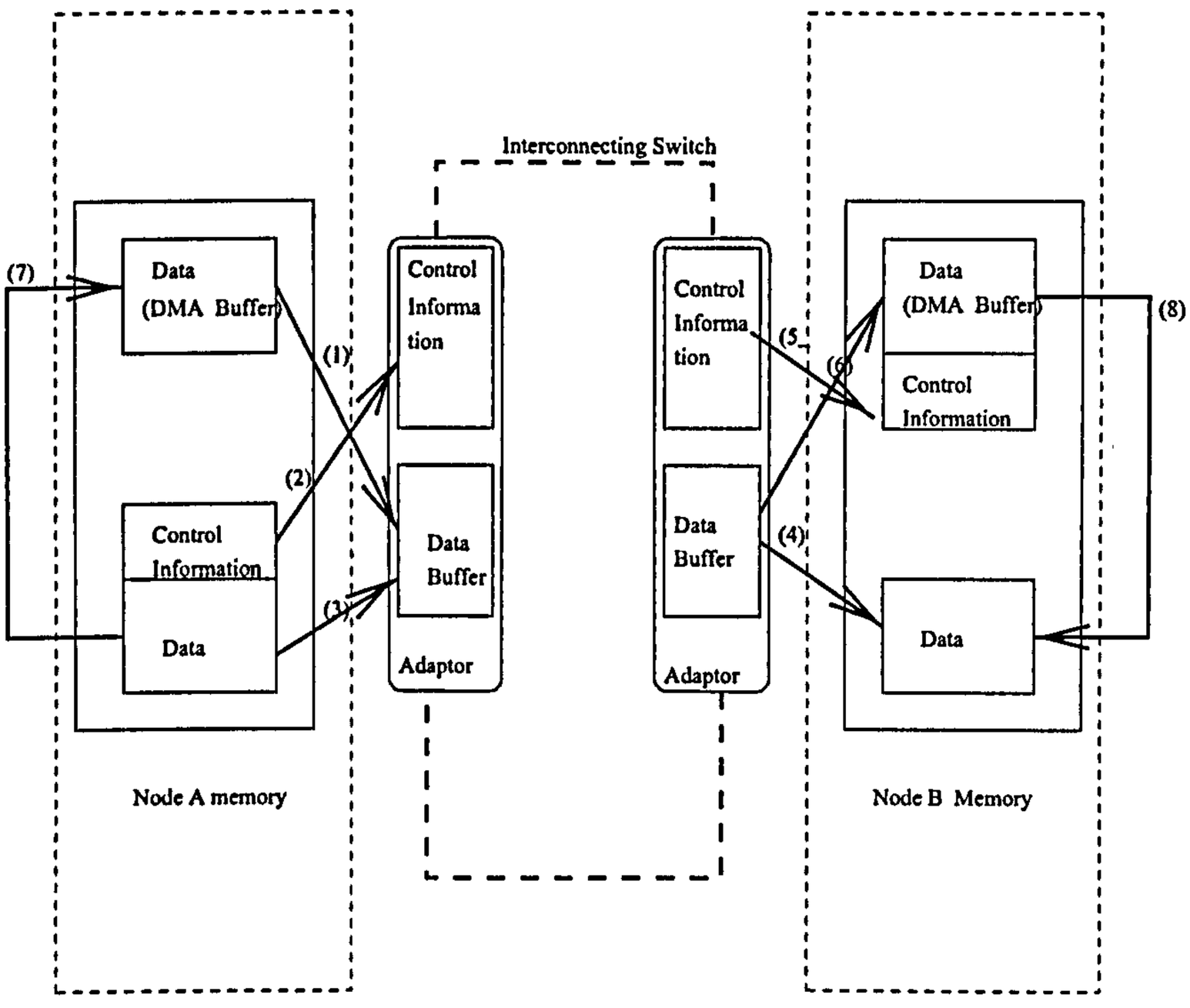


Figure 8.1: A message passing setup between two Nodes

(PIO) design is based on the assumption that the cluster interconnect interface is basically mapped onto the address space of a user space application. Issuing a user data transfer to the cluster interconnect interface is accomplished via initiating copy as well as control information operations to the cluster interconnect. In a second step, the cluster interconnect basically transfers the data to the receiver. For a send operation, this mechanism is outlined through bullet numbers 2 and 3. A PIO based approach is normally not being used on the receiving side as the transfer rates are for most cases unacceptably low(Araki 1998a, Bhoedjang 1998a).

A PIO based approach is being used in user-level communication sub systems such as FM (Warschko 1998a)or LFC (Bhoedjang 1998a). A Copy–DMA based design separates data transfers from control sequences. More specifically, control protocol data is submitted directly to the cluster interconnect utilizing PIO, whereas the actual data is memory copied first into a DMA–capable memory segment and in a second step the data is transferred using DMA method by the cluster interconnect. Copy–DMA is the technique that is being used by most LAN or WAN communication sub systems. Incorporating a slight modification that allows bypassing the protection boundary between kernel and user space memory, Copy–DMA is being used in communication sub systems such as Generic AM (Eicken 1992)or PM(Tezuka 1992). A Direct–DMA based design requires that the user space data that has to be sent resides in DMA-capable memory segments. The Direct–DMA based send technique avoids the memory copy between user and kernel space by allocating DMA buffers when the application is launched. It is important to point out that PIO and Direct-DMA provide a message-passing model, whereas Direct-DMA is based on a memory–mapped communication model. SCI (an IEEE Standard) or VMMC (Prylli 1997) incorporate a Direct–DMA based approach. The V2P–DMA technique substitutes bullet number 3 in figure(8.1) with a DMA operation. To accomplish that, a virtual to physical address translation (V2P) is required. The selected pages are pinned in memory to prevent the page-stealer from replacing these pages during a DMA operation. After the DMA transfer is completed, the pages are either surrendered to the memory sub system or may remain in a pinned state which allows reusing the pages (avoiding a second address translation).

A V2P–DMA technology is being incorporated into BIP (Prylli 1997) and VMMC–II (Dubnicki 1998), respectively. A PEX–DMA based communication mechanism is similar to V2P–DMA, but the pinned memory pages are marked as copy-on-write. This results in duplicating the pages if the application modifies the page content during a DMA transaction. The ramification is that the DMA operation always transfers accurate data. After the DMA operation, the copy–on–write flag is consulted to determine if the pages have been duplicated, and in that case the pages are surrendered to the memory sub system pool. Otherwise the flag is cleared, allowing the application to reuse the pages. A PEX–DMA based approach is being used with ParaStation2 (Warschko 1998a). In a PIO based approach, an application can safely reuse its send buffers as soon as the send operation finishes. With a Direct–DMA or V2P–DMA technique however, the application has to determine if a send buffer can be reused (via a simple DMA query operation). Receiving messages with Copy–DMA, Direct–DMA, or

V2P–DMA is similar to the send mechanism, except the data flow is reversed. On the sender side, PEX–DMA is considered as being a refinement of V2P–DMA. On the receiver side, PEX–DMA is a clear improvement over Copy-DMA.

8.5 Software protocol model

Over the last few years, the increased study of protocols for message passing based cluster systems represents a significant shift from the traditional approach normally taken to solving problems of computer system performance. In other words, the shift of focus can be described as a process away from performance behavior or product behavior of data communication systems towards the study of the state and structures of the protocol that enter into the behaviour. While focusing on bandwidth performance, the central focus revolves around the structure of the communication protocol that optimises the overall communication bandwidth, and increases the utilisation of the I/O ports as well as the network channel adapters in the most efficient and effective way. The potential performance limiting components are (1) the software memory-to-memory copy rate, (2) the Node I/O bus bandwidth, (3) the adapter's mean processing rate per packet and (4) the switch bandwidth. We believe that the three fundamental questions that arise in bandwidth optimisation are, (1) what constitute the weakest link among the four paths, (2) what features of the protocol design lead to this performance results, and (3)how can these features be effectively improved.

As we pointed out earlier, adaptor has the purpose of processing the message passing control information and data to make the communication protocol software on the node receive and transmit messages with other nodes within the cluster. Data is inserted into the switch by a server(node) through the switch adaptor. The adaptor is the hardware device between the server and the switch. An adaptor is connected to the switch chip and the node's internal I/O bus.

Increasingly, an adaptor has become more sophisticated in its decision making functions. It interacts with the operation system software protocol. The later protocol takes a request from a user space application of message size (msg) Bytes, breaks this into a series of packets of sizes $2K$Bytes. These packets are then provided with sequence numbers for identification. The entire message is now put in pinned memory on the node for the adaptor to read using a DMA operation. In the IBM InfiniBand switching technology these messages are Work Queues (WQs). These data WQs are transferred from the node through the I/O bus in both

directions using DMA, where the size of a DMA transfer is 128 Bytes. On the IBM RS/6000 implementations of the InfiniBand switch there are several independent software control structures called *Queue Pairs* that send or receive data to and from the switch. A send Queue Pair has its own send data buffer, a memory region on the node, called send FIFO, from which the adaptor does the DMA read. The receive Queue Pair has its data buffer too.

Ensuring the overall balance in data communication between the main memories of the servers and the switch, protocol design must include a flow control. The easier way of doing this is to set a maximum threshold size N for the number of packets in the send queue so that if the number of packets ever reaches that level then the protocol stops sending more packets and must wait. Like TCP/IP protocol, the number of packets that can be sent before waiting for acknowledgement is called a *window.* In our modelling, the value of this threshold is the size of the buffer queue in the $M/G/1/N$ framework.

In this work, we model a set up of this in which messages flow from a sender node to a receiver node. The objective is to predict in transient time the queue length, representing the number of packets that are still in the pipeline that has not been acknowledged.

After studying the inter arrival times of MPI requests for a typical application benchmark, called **NAS,** we noticed that a sender node send message requests at some Poisson rate of λ (i.e. λ requests per second) to the receiver node. Because each packet is provided with a sequence number, the receiver sends acknowledgement of that packet with its sequence number back to the sender protocol. When the sender protocol gets the acknowledgements for each packet it removes that packet from its queue of untransmitted packets. To illustrate this, let n_t be the number of outstanding packets in a protocol queue. Also let $S_1, S_2, S_3, \ldots,$ be a sequence of inter arriving times of acknowledgements. Define $T_1 = S_1, T_i = S_1 + S_2 + \ldots + S_i$ for $i \geq 2$ and for $t \geq 0$. Packets depart the protocol queue at random times $T_1, T_2, \ldots,$ the times when the protocol receives the acknowledgements and removes the corresponding sequence numbers from the sender queue.

We create a two dimensional process $x_t = (n_t, \tau_t)$, where n_t is the number of packets in the sender queue as before and τ_t is the time since the last acknowledgement. The process $x_t = (n_t, \tau_t)$ is still not in the PDP form because of the following reasons:

1. Although the message arrives at Poisson instances, the message lengths $X_1, X_2, X_3, \ldots$, are independent random variables having a general common probability distribution G with density g and mean $\hat{X} = E[X]$. This distribution needs to be determined.

2. The rate u at which these messages complete transmission, gets across the communication link from sender to receiver, needs to be known before a queue length model can be constructed.

3. The sequence numbers are per packet and not per message.

As things stand, it is not possible to put things in the correct Markov form, and to make the problem tractable, we have to conduct some experiments to determine the distribution of message sizes and the link bandwidths. The easier way to do this is to run a MPI Benchmark, collect the data and use data analysis techniques to estimate G. This approach is described in section(8.6). In that section we will show how the ideas of estimating parameters can be used to identify the general distribution from collected data. This is useful in a number of ways: first, practical MPI problems can be solved by applying the approach, and second, we identify the appropriate form of the model. Software protocol is responsible for the control of flow of messages as they are transmitted from the sender node to the receiver node at a constant rate of $u > 0$ whenever the inventory of the untransmitted packets in the protocol *window* is positive. The concept is similar to the flow control window used in TCP/IP protocol, where at any instance, the sender maintains *sequence numbers* corresponding to packets that are in the pipeline of the link and have not been transferred and acknowledged. It will be realistic for the bandwidth u of the link and the average message size $E[X]$ to satisfy the stability condition $u > \lambda E[X]$, that is the transmission rate through the I/O bus and the adaptor rate is larger than the average amount of packets offered to the protocol window per unit time. Thus, we are interested in the time dependent inventory of untransmitted packets in the protocol Window.

The easiest way of synchronising a message size and the sequence number is to introduce an extra ingredient into the design of the protocol acknowledgement sub system: tune the protocol to provide an acknowledge to the user application only when all the packets associated with the message have been acknowledged.

After measuring the sizes of messages and the sizes of the inter departure times of these messages we found that the distribution of ratio of message size and bandwidth X/u is roughly the same as the distribution of the inter departure

times. We model the system with a range of values of bandwidth $u > 0$. Our based time is 1 micro second. A typical bandwidth is $1.3GB/sec$. This is equivalent to a transfer of $1.3K$ Bytes every micro sec $= 1.3KB/\mu sec$[1].And since the protocol uses a $2KB$ per packets, this translates to 0.6 packet every micro second.

8.6 Estimating distribution of message length

We run an MPI NAS benchmark on an existing cluster of servers, varying the number of tasks running from 8 to 256. This data set only represents NAS MPI calls to send Bytes of messages across one server to the another. It does not explicitly account for control packets sent by the protocols (ACKs, rendez–vous handshakes). Although this data should not be interpreted as a distribution for scientific and technical computing, it does represent a typical distribution of MPI messages.

[1]Assume that $\mu = 10^{-6}$.

Sizes	*counts*
0 – 1	185695
2 – 3	0
4 – 7	22620
8 – 15	434428
16 – 31	15880
32 – 63	3505
64 – 127	7589
128 – 255	151423
256 – 511	446148
512 – 1023	523562
1024 – 2047	409528
2048 – 4095	149302
4096 – 8191	152076
8192 – 16383	131956
16384 – 32767	179553
32768 – 65535	174427
65536 – 131071	203531
131072 – 262143	155965
262144 – 524287	93626
524288 – 1048575	13056
1048576 – 2097151	7630
2097152 – 4194303	2024
4194304 – 8388607	154
8388608 – 16777215	968
16777216 – 33554431	0
33554432 – 67108863	308
67108864 – 134217727	0
134217728 – 268435455	0
268435456 – 536870911	0
536870912 – 1073741823	0
1073741824 – 2147483647	0

Each benchmark will be different. . The MPI/LAPI queue model will be incomplete if we do not work out and estimate the distribution of the message lengths. To help us towards this goal, we have run a NAS experiment and got the above results. Here, we can take it that the total count $T = 3279259$ is just the sum of all the counts on the right hand column. If we eliminate the first two

columns and replace them with a new column=(First column+second column)/2. We then ehave two columns with the first few terms as follows:

Sizes	counts
2	0
4	22620
8	434428
16	15880
32	3505
64	7589
128	151423
256	446148
512	523562
1024	409528
2048	149302
4096	152076
8192	131956
16384	179553
32768	174427.

We can approximate this distribution with all kinds of distribution: exponential, Erlang, Gamma, Pascal or a discrete distribution. Let f_0 be number of counts whose length is greater than 0 byte, f_2 number of counts whose length is greater than 2 bytes, f_3 is the number of counts whose length is greater than 4 bytes, etc. Of course, $f_0 = T = 3279259$, leading to a new matrix which looks like this:

x bytes	total counts $> x$ bytes
0.0000	f_0
1	f_1
2	f_2
...	...
x	f_x.

By arranging this sequence of recorded (x, f_x)(message size, count $> x$) we have table of vector entries (x, f_x), in decreasing length. From this we can estimate the probability distribution of message sizes, similar but more refined than that shown in (8.2).

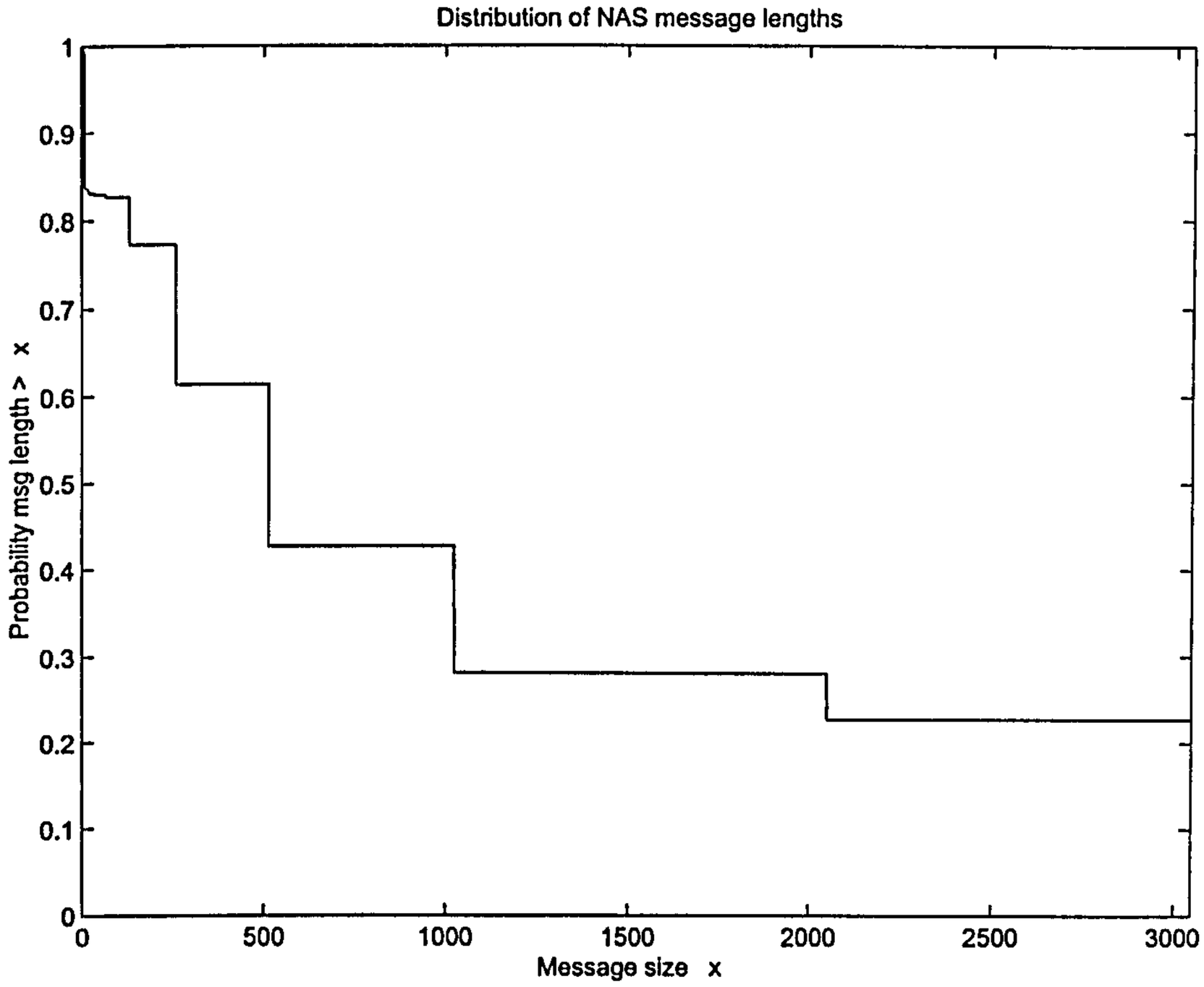


Figure 8.2: A smal example of a discrete distribution

8.6.1 Computing the Harzard rate

To compute the Harzard rate, we create a long vector R of dimension f_0. This is a sample of message sizes from f_0 discrete trials and we might assume that it is an independent sample from some given density function with some parameters that need to be estimated. We can assume that it is exponential data with mean $\hat{X} = 1/\mu$. We discovered that $\hat{X} = 4k$Bytes with 95% confidence interval $[4095, 4108]$. We use a Gamma function with density

$$g(x/\alpha, \mu) = \frac{\mu^{\alpha} x^{\alpha-1}}{\Gamma(\alpha)} e^{-x\mu},$$

where $\Gamma(\alpha)$ is a complete Gamma function. The optimisation for estimating the parameters α and μ becomes much difficult and takes a very long time to complete, producing the results $\alpha = 0.3270$ and $1/\mu = 12542$Bytes with confidence intervals $[0.3261, 0.32794108]$ and $[12503, 12581]$, respectively. We can now compute the Harzard rate as we explained in the previous chapter. Taking note $\hat{X} = 12542 * 0.3270 = 4101$ Bytes is the average message size which is the same as that of exponential distribution. But the variance of the message sizes of this Benchmark is pretty large. The coefficient of variation is $1/0.3270 = 3.1$, the ratio

of the standard deviation and the mean. This is the measure of the message sizes. This is large indicating that the message sizes actually varies more than an exponentially distributed random variable.

8.7 PDP model for software protocol communication

Let n_t represents the number of packets in the system at time t. With this, one can define a natural filtration $\Im_t = \{n_s : s \leq t\}$, obtaining the process $\{n_t, \Im_t\}$, which, of course, is not a Markov process.

In this uni–direction study, protocol on the sending node receives acknowledgements (ACKs) of packets that have arrived at the receiver node. The receiver node only sends ACKs when the data is copied from the pinned memory to the user memory. For instance, let $\ldots, S_{i-1}, S_i, S_{i+1}, \ldots$, be the renewal times of the ACK process with inter–ACK—time distribution G. We are claiming here that the arrival of ACKs in the sender process is a renewal process, where S_i denote the last ACK–time prior to some fixed time t, then the residual inter–ACK–time $S_{i+1} - t$ depends on the filtration $\Im_t$. To be more specific, this depends on the time $\tau = t - S_i$ since the last ACK arrival.

Using this construction, we can predict the future from the past in the following way. Let $x_t = (n_t, \tau_t)$, where τ_t is the time that the sender node has already been waiting without packet being ACKed, and n_t is the number of packets which protocol have in the pipeline. In fact, the following inequality must hold

$$(1 \leq n_t \leq N, 0 \leq \tau_t \leq D),$$

where N is the total maximum number of packets that can be allowed in the system at any given time and we choose D so large that it will be impossible to stay up to time D without an ACK arriving. In our implementation, $N = 64$ packets.

In addition, consider a modified time–dependent process with state $X_t = (x_t, t) = (n_t, \tau_t, t)$. The $\{X_t\}$ is a Markov process according to Davis. Define a finite horizon cost in the time interval $t \leq l \leq T$:

$$V(x) = E_x \left[\int_0^T f(x_l, l) dl + \Omega_T(\tau_T) \right].$$

Here, the expectation E_x is taken over the finite time interval $[t, T]$, conditioning on the state $x = (k, \tau, t)$. But we carry out the transformation $t = T - t$, so that the boundary time at $t = T$ becomes the initial time at $t = 0$. Then, we look at the process at time interval $[t = 0, t = T]$.

1. For $k = 0, n_0 = 0$, there are no packets in the system. Therefore, there is no departure at time at the neighbourhood of $t = 0+$.

2. For $t < 2L$ and initially at $t = 0, n_0 = 0$, where L is the one–way latency, there is no departure.

3. For $t \geq 2L$, $k = 2, \ldots, N-1$:

$$\left.\begin{array}{rcl} \frac{\partial}{\partial t}V_k(\tau,t) & - & u\frac{\partial}{\partial \tau}V_k(\tau,t) \\ & + & \beta_k V_{k+1}(\tau,t) \\ & + & \gamma_k(\tau)V_{k-1}(0,t) \\ & + & \alpha_k(\tau)V_k(\tau,t) \\ & = & f_k(\tau,t). \end{array}\right\} \tag{8.7.1}$$

This is a discrete version of the algorithm in the backward form.

$$pV_{k+1}(n+1,t+1) + q_n V_{k-}(0,t+1) \tag{8.7.2}$$

$$- r_n V_k(n+1,t+1 - V_k(n,t) = 0, \tag{8.7.3}$$

$$k = 2,3,\ldots,N, n = 1,2,\ldots,t = 1,2,\ldots, \quad t \leq \tag{8.7.4}$$

$$\tag{8.7.5}$$

where p is the Binomial distribution parameter and q_n is the discrete Harzard rate as in chapter (7).

4. For $k = N$: **The protocol queue buffer is full** and no packets is admitted into the system. $\lambda = 0$.

5. For $k = 0$: or $t < 2L$ **no departure occurs:**
$\frac{d}{dt}w(t) - \lambda V_1(0,t) + \lambda w(t) = f_0(t)$,where
$w(t) =$ value of V at $k = 0$, Poisson rate $= \lambda$, Hazard rate of G $=$
$\mu(\tau), \alpha_k(\tau) = \lambda + u\mu_k(\tau), \beta_k = -\lambda, \gamma_k(\tau) = -u\mu_k(\tau), \beta_N = 0$, for$t > 0, k =$
$1, \ldots, N, \tau \in [0, D]$, and boundary conditions:
$V_k(\tau, 0) = \Omega_k(\tau), w(0) = 0, V_k(D,t) = V_{k-1}(0,t), V_1(D,t) = w(t)$.

8.8 Upwind scheme for protocol queue model

Let $t = T - t$. Then, we have,

for $k = 2, \ldots, N-1$:

$$\begin{aligned} \frac{\partial}{\partial t}V_k(\tau,t) &- u\frac{\partial}{\partial \tau}V_k(\tau,t) + \beta_k(\tau)V_{k+1}(\tau,t) + \\ \gamma_k(\tau)V_{k-1}(0,t) &+ \alpha_k(\tau)V_k(\tau,t) = f_k(\tau,t). \end{aligned}$$

Let $V_k = V, Y = V_k(0,t)$ and

$$\frac{dV}{dt} \approx \frac{V_m^{n+1} - V_m^n}{h},$$

$$\frac{dV}{d\tau}$$

can be defined in 3 ways:

$$\frac{V_{m+1}^n - V_m^n}{h_\tau} \tag{8.8.1}$$

$$\frac{V_m^n - V_{m-1}^n}{h_\tau} \tag{8.8.2}$$

$$\frac{V_{m+1}^n - V_{m-1}^n}{2h_\tau}. \tag{8.8.3}$$

(8.8.4)

$$\frac{V_m^{n+1} - V_m^n}{h} - u\frac{V_{m+1}^n - V_m^n}{h_\tau} + F(Y_1^n, f_m^n) = 0, \tag{8.8.5}$$

where $F(Y_1^n, f_m^n)$ is given.

$$\begin{aligned} V_m^{n+1} &= \left(1 - \frac{uh}{h_\tau}\right) V_m^n \\ &+ \frac{uh}{h_\tau}V_{m+1}^n - hF(Y_1^n, f_m^n). \end{aligned}$$

The Upwind algorithm implemented in Matlab:

1. **Set parameters:**

For $i = 1 : M+1$
  For $k = 1 : N$
    compute $\mu, \alpha(k,i), \beta(k,i), \gamma(k,i)$.

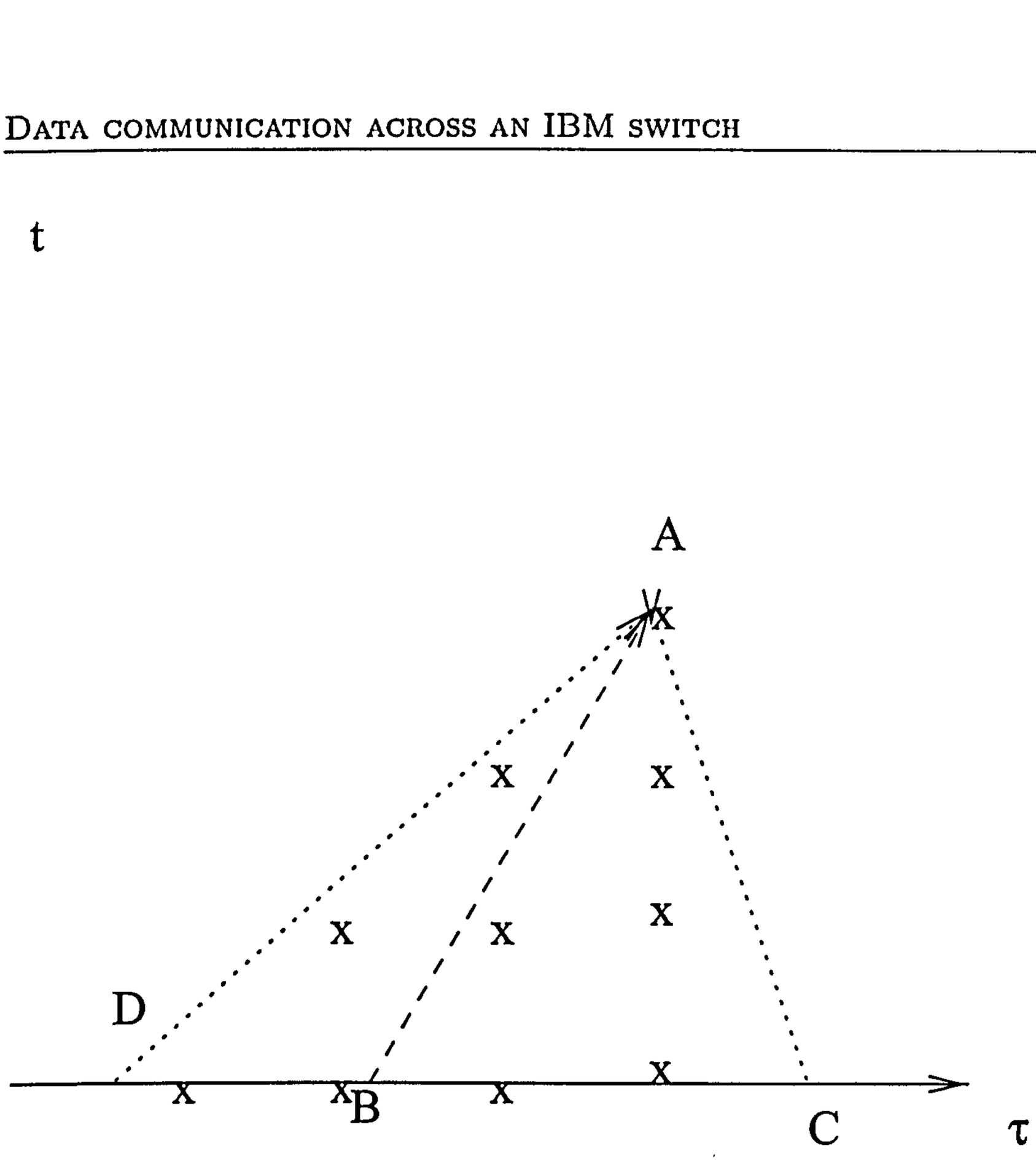


Figure 8.3: Courant CFL condition

2. **Initilisation:**

$$\begin{aligned} \text{For} \quad i \;=\; & \quad 1 : M+1 \\ & \text{For} \quad k = 1 : N \\ & \qquad \text{compute } V(k,i) = k - 1. \end{aligned}$$

3. **Repeat: Start from j=1 until** $j * h > T$ **:**

$$\begin{aligned} &\text{While } (j * h \leq T) \\ &\text{For} \qquad\qquad i \;=\; 1 : \; M+1 \\ &\qquad\qquad\qquad \text{For} \quad k \;\; = 1 : N \\ &\qquad\qquad\qquad\qquad \text{compute new } V \\ &\qquad\qquad\qquad\qquad V_m^{n+1} = \ldots \end{aligned}$$

4. **Update** V **and go to Repeat, else goto Exit**

5. **Exit.**

As shown in figure (8.3), the value of V_m^{n+1} depends on data given in the triangle with vertex (τ_m, t_{n+1}) and on the data at points initial line

$$\tau_{m-n-1}, \tau_{m-n}, \ldots, \tau_{m-1}, \tau_m.$$

V_m^{n+1} depends on data given at points on the triangle. Thus, the triangle is called the *domain of dependence* of V_m^{n+1}.

The Courant CFL condition restricts the possible choices of step sizes h and h_τ, since $u > 0$ CFL provides an upper bound for the size of the time step. Therefore, the CFL condition provides the necessary condition for stability. It is not a sufficient condition. Figure (8.3) illustrates two situations in which the CFL condition is violated. These occur when the characteristics DA or AC lie outside the triangle.

8.8.1 Results of packet length estimation

We assume here that the reader has at least some knowledge of the more general probability distributions like Gamma, Erlang(those introduced in the previous chapters with harzard rate). Several ideas from these chapters are used in the construction of the Matlab codes. For general distribution G, we choose an Erlang distribution and set D to be large enough. This modification approximates $M/G/1/N$ so well. In addition, our reader should also note that a packet is $2k$ Bytes and all time in x–axis of our graphs are all measured in *micro seconds,* not in seconds. Below are the results of the queue lengths compared with those obtained through stationary values of Erlangian queue system. We solve the equations with Matlab. The functional call to the Matlab code developed is $que - -Lapi(\lambda, \mu_0, u, D, N, N2, \phi_0, T, M, k)$, where

- λ is the Poisson rate.
- μ_0 is Erlang service parameter with mean holding time (k/μ_0), where is an Erlang parameter defined below.
- u is the link bandwidth.
- D is the truncation discussed in the previous chapters.
- N is the buffer size.

- $N2, \phi_0$ are not used here. They are important in other PDP data communication applications when the buffer size has an upper boundary N and a lower boundary $N2$.
- T is the terminal time.
- M is the number of steps in the finite element scheme developed.
- k is a parameter in the Erlang model that shape the distribution function increasing the mean holding time when it increases, see (Morse 1958).

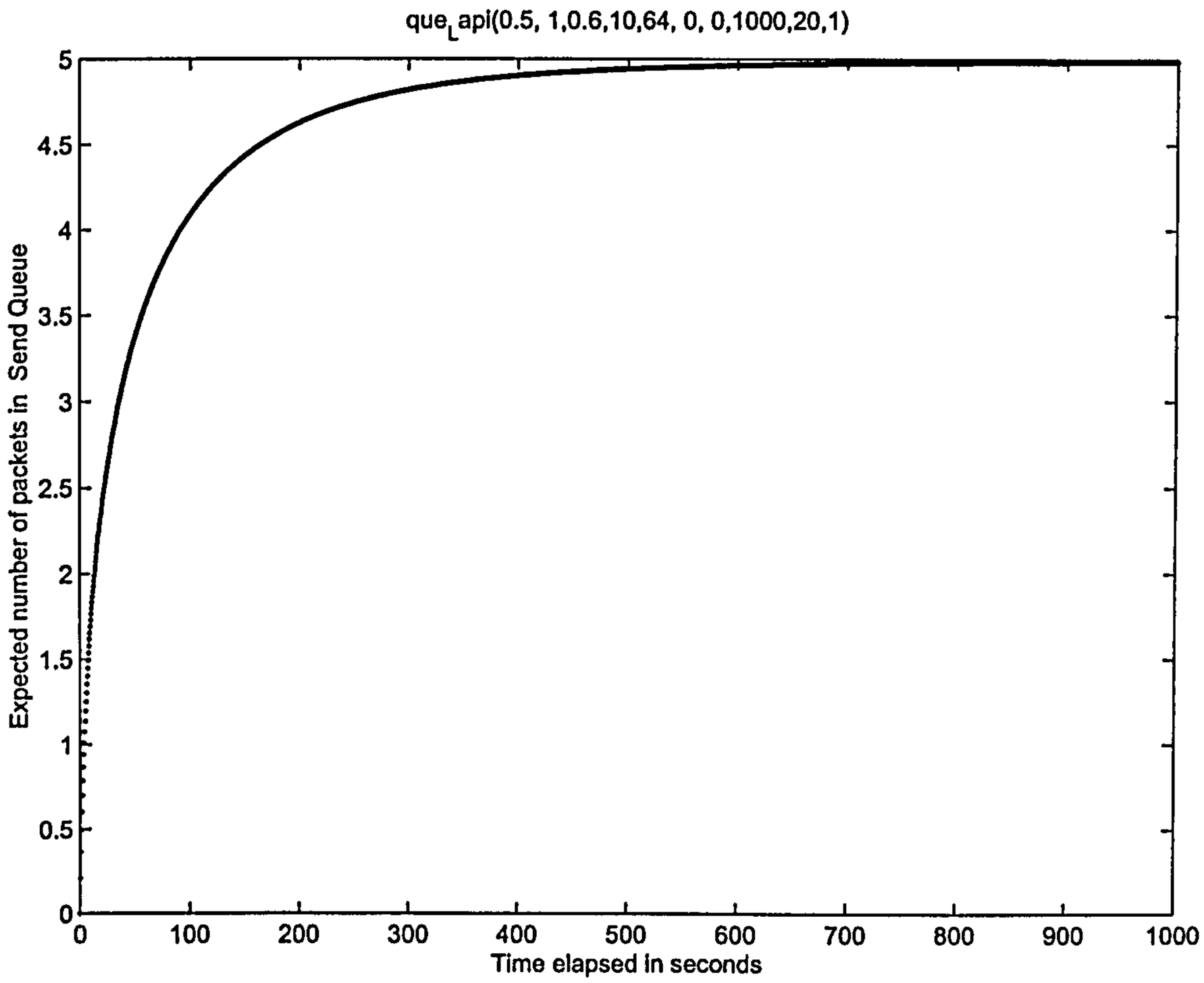


Figure 8.4: Number of packets in system(in micro seconds)

In this experiment, we assume that there are zero packets in the system initially. We also simulated this with a range of values of bandwidth $u > 0$. Our based time is 1 micro second. A typical bandwidth is $1.3GB/sec$ which translates to $1.3KB/microsec$. And the protocol uses a $2KB$ per packets, this translates to 0.6 packet per micro second. Therefore $u = 0.6$ packets per base cycle or base time. We also need to translate the distribution of message lengths from the units of Bytes to units of packets. All we need to do is divide everything in $2KB$ per packet, which convert the units from Bytes to packets. The following graphs show the results of the matlab simulation:

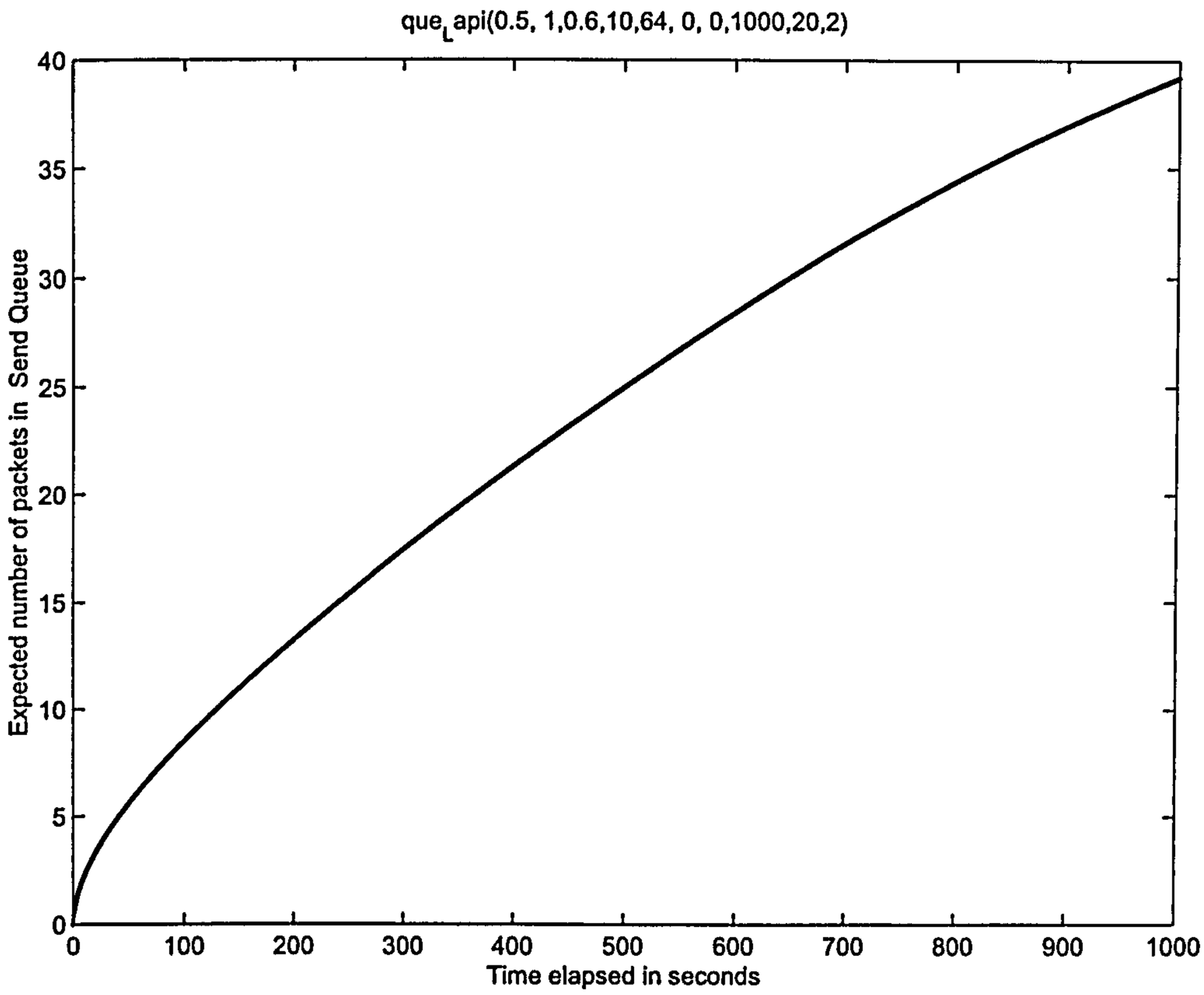


Figure 8.5: Number of packets in system(in micro seconds)

1. The code entry "que–Lapi(0.5, 1,0.6,10,64, 0, 0,1000,20,1) " produces figure(8.4) with steady state mean of 4.9819 packets. $\lambda = 0.5$ packets per second=1K Bytes of data per second. Since $k = 1$ in the Erlang system and $\mu_0 = 1$, a single packet is transmitted every second = a block of $2k$ Bytes leaves the system every second. Therefore, the offer load is 0.5

2. The code entry "que–Lapi(0.5, 1,0.6,10,64, 0, 0,1000,20,2) " produces figure(8.5) with no steady results after 1000 micro seconds(not seconds). $\lambda = 0.5$ packets per micro second=1K Bytes of data per micro second. Since $k = 2$ in the Erlang system and $\mu_0 = 1$, mean inter departure time is $k/\mu_0 = 2$, meaning that on average, a single packet completes transmission every 2 micro second. Therefore, the offer load is 1. Notice that the system did not converge after 1000 micro seconds as in the case in figure(8.4). This is because the offer load is now unity and transient persist since after 1000 micro seconds.

3. The code entry " que–Lapi(0.5, 1,0.6,10,64, 0, 0,2000,20,2) " produces figure(8.6) with no steady state. The entries here are the same as those in figure(8.6) except that we have increased the time interval to 2000every 2

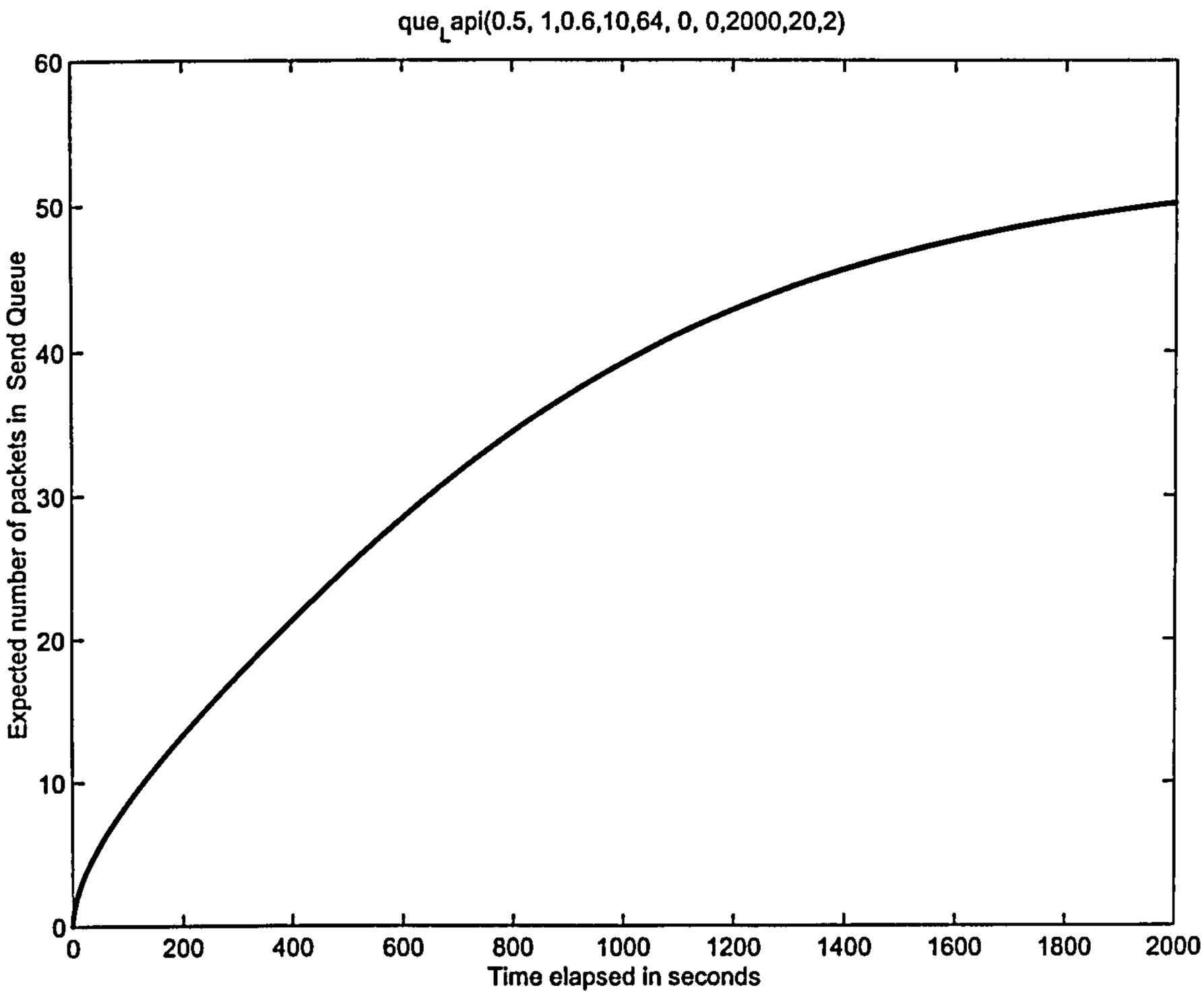


Figure 8.6: Number of packets in system(in micro seconds)

micro second. Therefore, the offer load is 1. Note also that the system did not converge after 2000 micro seconds as in the case in figure(8.4), but the amount of queue length of packets remain below 64, the size of the Send Queue buffer.

4. The code entry "que–Lapi(0.5, 1,0.6,10,64, 0, 0,2000,20,2) " produces figure(8.7) with steady state 52 packets after 4000 micro sec.

Other steady results are shown in the following table.

Parameters	Results
$que(\lambda, \mu_0, u, D, N, N2, \phi_0, T, M, k)$	Mean number of packets
$que_Lapi(0.5, 1, 0.6, 10, 64, 0, 0, 1000, 20, 1)$	4.9819
$que_Lapi(0.5, 1, 0.7, 10, 64, 0, 0, 1000, 20, 1)$	2.4946
$que_Lapi(0.5, 1, 0.6, 10, 64, 0, 0, 50, 20, 4)$	4.5198
$que_Lapi(0.5, 0.6, 0.6, 10, 64, 0, 0, 100, 20, 1)$	1.0000

Also see figure (8.8) for transient solution profile. We found that after a very long

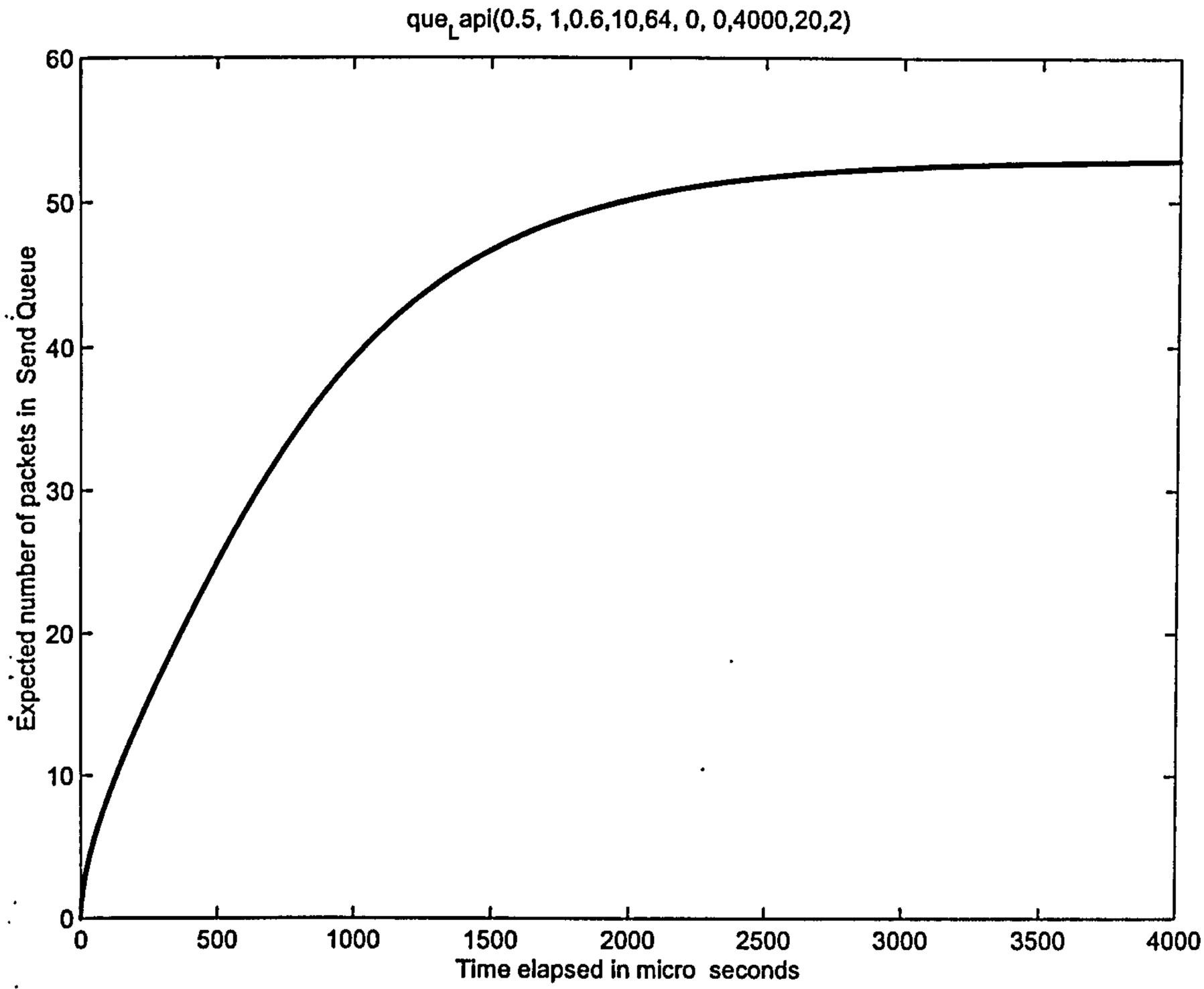


Figure 8.7: Number of packets in system(in micro seconds)

time the system converges to a stationary distribution irrespective of the initial number of packets in the system.

8.9 Estimating and optimising link bandwidth u

The answer to the first question is often based on the copy rate of the communication software, an actual measure of the rate at which data is transferred into the pinned memory area out of the user buffer (on the send side), or the rate at which data is moved onto the user buffer out of the pinned area on the receiver side. The answer to the second question can be formulated by the specification of the host hardware architecture, the systems support for CPU load and store operations, along with how the system's principles interact with the software communication code to yield a particular memory-to-memory copy operation rate. Although a copy rate is the result of a software operation, its actual cause is the state of the hardware structures involved, and the speed at which the system is capable of storing and loading data to and from the memory sub system.

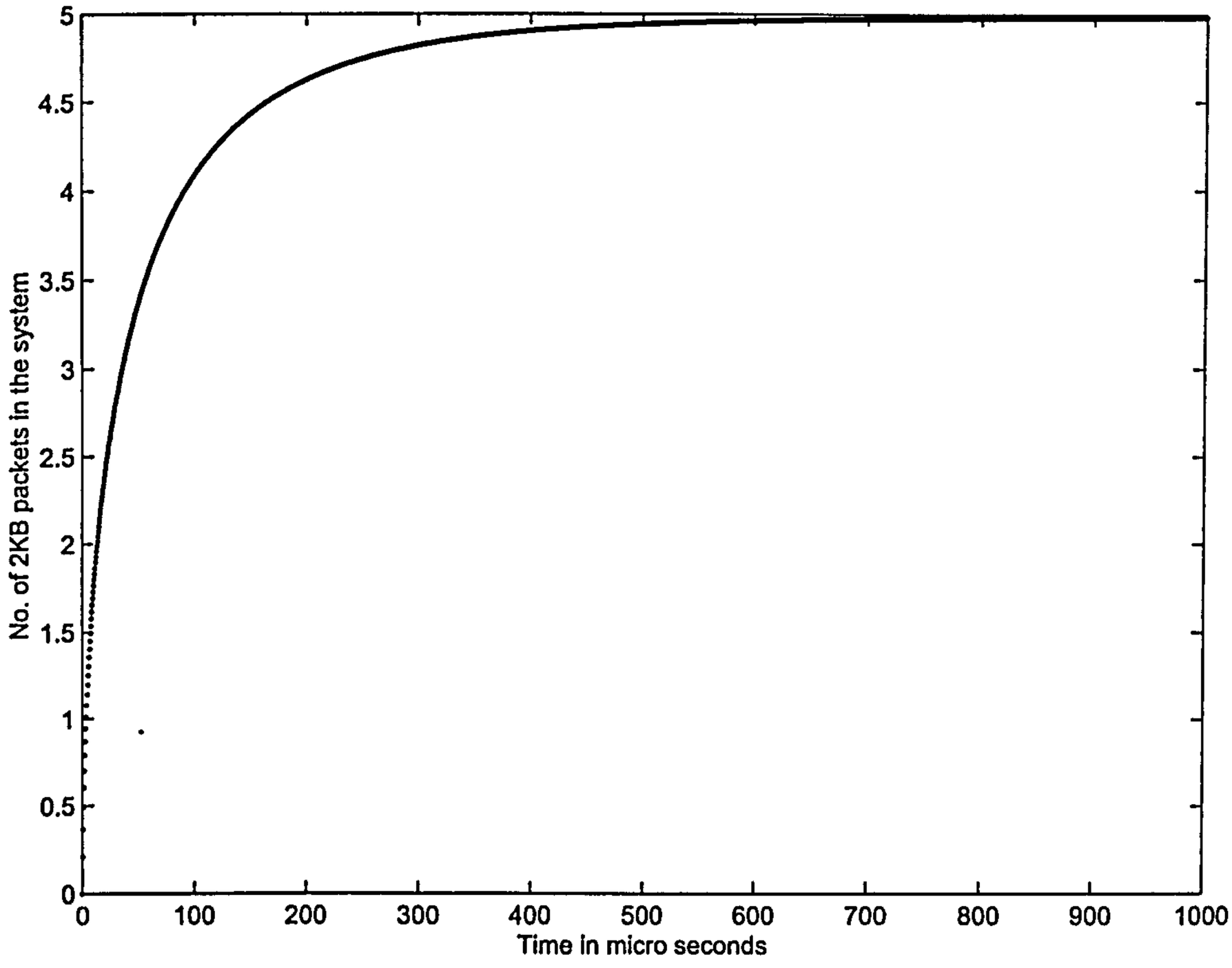


Figure 8.8: No. Packets in flight

Although the IBM RS/6000 Scalable Power parallel (SP) super computer is now more than eight years old, and enjoying the third generation of switch technology, the SP switch is still quite a mystery to most computer science researchers working on performance analysis and modelling. It embodies leading–edge technology in an area where few modellers have access to.

Here, we report the application of Davis theory to the development of a set of modelling tools for the study of message passing in IBM SP and InfiniBand switches. SP and the recent InfiniBand adaptors evolved from the use of computing servers connected by a switch. The system comprised of AIX(Unix) and Linux servers grouped together physically in building blocks(frames).

This cluster enables parallel applications such as MPI to run on these servers. MPI is a programming software package with an environment that allows an application programmer to write parallel intensive computing codes that are logically structured without taken into consideration the hardware topology of the cluster.

We shall analyse and model this hardware part of data communication under a zero–copy protocol condition. Therefore, the memory to memory copy from the user address space to the kernel address space on the send side is eliminated and

the copy from kernel to user space on the receive side is also eliminated. The most important component for optimising bandwidth is the I/O hardware port as is illustrated in figure(8.9) for an infiniBand adaptor on an IBM server. A user task, running MPI, initiates a I/O request on Node A to submit a message to Node B. With zero copy communication, data is transferred straight from user (virtual address) into the send adaptor adapter to transfer through a DMA read operation. A similar (but reverse) process is used on the receiver side through a DMA write operation. During this investigation, we realised that the DMA Read bandwidth is higher than the DMA Write bandwidth. So, we seek an optimisation aimed at improving Write bandwidth. In the hardware message are transmitted in cache line basis using direct memory access (DMA) operations through the I/O port and the adaptor hardware.

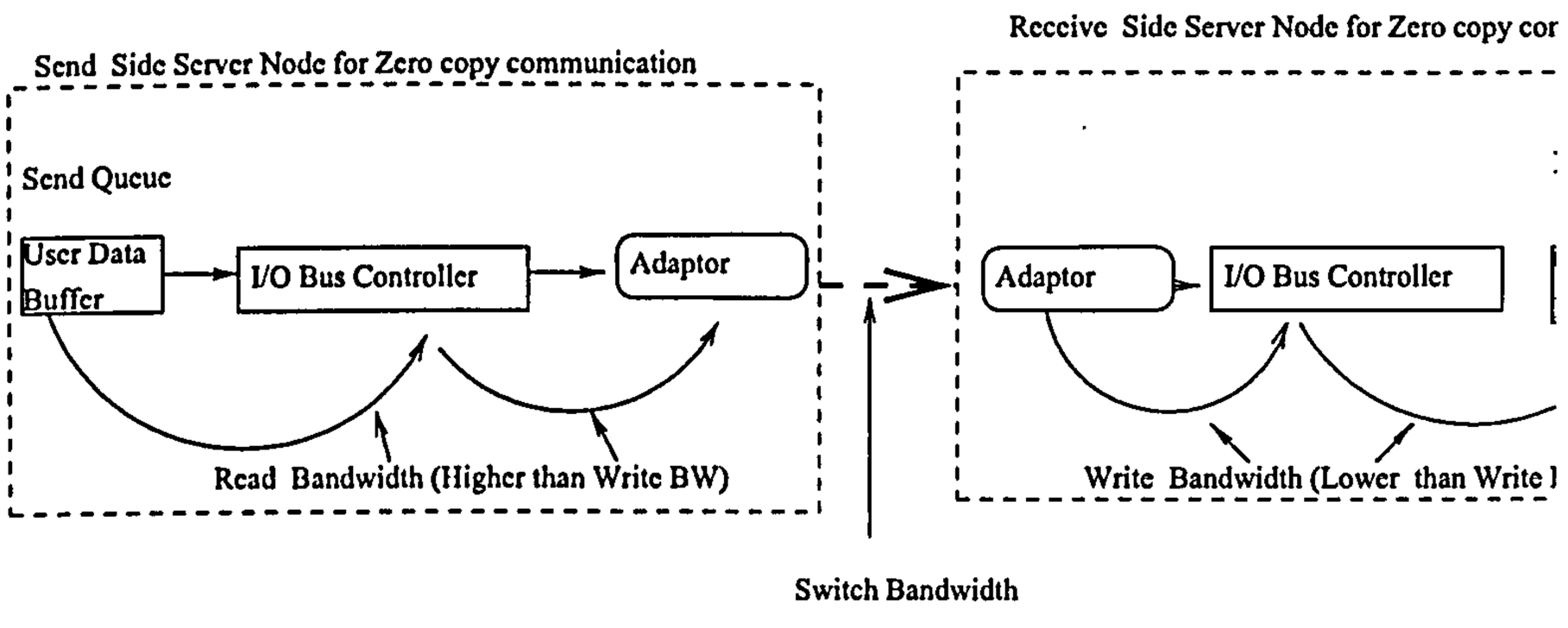


Figure 8.9: DMA traffic paths for zero copy

All commands from the adaptor go through an outbound I/O bus and the response signal is received in the inbound bus and is forwarded to the adaptor. A command from the adaptor such as DMA Read or Write command requires a small number of processor clock cycles (pclocks). If the DMA Read command is accepted the return data will come back later in the inbound bus. There are two major commands that are important in this analysis: the DMA Read commands (the commands that require data to be returned from the memory sub system, and the DMA Write commands (commands that immediately followed data). Data transfer occurs in cache line basis, which is 128 Bytes.

The purpose is to predict the DMA write bandwidth. There are three write buffers in this I/O port. Multiple adaptors are to be connected onto an inter–connected bus similar to a switch within a computer node which connects I/O and CPU units into the memory sub system. In recent years there have been a resurgence of

interest in *ring* communication topology for this inter–connected bus, connecting various server modules together. Each module consist of several CPUs in units called cores, I/O ports and memory sub system units. Since in a ring communication topology some modules are not directly connected, cache lines of DMA and CPU request data will have to hops along intermediate modules until they arrive at their final destination modules. In recent years, IBM server development for the high end scientific applications has used this topology, departing from their previous designs that used shared lines on a single bus. Unlike shared lines, a ring is capable of simultaneous data transfer. Here the first module can send to the second and at the same time as the second can send to the third, etc.

A shared bus communication interface is a major bottleneck, limiting DMA bandwidth through the I/O port. Designing bus system for optimising I/O bandwidth is very difficult. Maximum bus speed is often limited by physical factors like the bus length and the number of modules that are connected on it. To lower cost IBM previous shared bus design used a single bus, called the 6XX bus, for connecting the CPU units, the I/O and the memory sub systems. This shared bus made use of some features in PCI and ATM technologies that utilising a combination of shared memory coherency protocol for data consistency among modules and I/O bus protocol for DMA transfer. In this design any I/O device like the adaptor or CPU on a CPU module can initiate a memory Read or a Write transaction. If there are many of such initiations, an arbitration scheme is required to decide who gets the bus next. This design limits the number of I/O devices that can be connected to the bus. Thus, limiting the maximum aggregate bandwidth for DMA operations.

In (Field, Harrison 1995) and other follow–up studies, a stochastic queue model was presented for modelling a ring–based shared memory IEEE P1296 Standard Coherent Interface(SCI). Unlike our model, their analytic queue model assumes a steady state condition. We analyse the traffic paths across an IBM server that uses the SCI ring within its server node and seek for the effective bandwidth at the various paths of the ring when the number of I/O adaptors connected to it increase from one to 8. These are the features that affect aggregate I/O bandwidth:

1. the inter arrival times of DMA traffics from the adaptor to the I/O ports. Hardware simulation using VHDL in the research and development indicated that this process is Poisson with mean inter arrival time of 110 pclocks, i.e. $\lambda = 1/110$ =1 job per 110 pclocks=$128/110 = 1.1636B/s$.

2. the lifetimes of the Write buffer(there are 3 of them) that hold DMA Write data before releasing the data onto the SCI ring. The mean lifetime of this buffer is a function of the utilisation of the SCI ring. In addition, the utilisation of the SCI ring is also a function of how many adaptors are connected onto that ring. We also found from hardware simulation that the life times follow a general distribution in such a way that the mean lifetime increases when the number of adaptors connected onto the ring increases.
3. We introduced an intermediate queue, called " Fabric queue"[2], at the memory side of each module en route to the memory that temporarily hold data from the 3 I/O buffer. In the first design, it took 330 pclocks to get a DMA write data to memory and get response back to release the I/O buffer holding the DMA data when only a single adaptor is connected on the SCI ring. And on a machine running at $1.2GH$ frequency, this will produce a unidirectional bandwidth of $(128/330) * 1200 = 465MB/s$. When we used PDP queue and introduced this Write queue that sends an acknowledge to the I/O controller whenever it pulls the DMA data off the SCI ring, the I/O buffering was able to reuse its buffer as soon as it gets the infomation that the data has been pulled off the ring. This reduces it mean lifetime from 330 pclocks to 220 pclocks when there is just a single adaptor connected on the ring, almost tripling the unidirectional bandwidth. The I/O bus accepts a DMA cache line into the system from the adaptor when one of the 3 buffers is no longer being used; thus a Write operation from the adaptor is said to be complete when the buffer that was temporarily holding the cache line data could be reused.

As described in this work, the sizes of the cache line DMA data are deterministic, each with value 128 Bytes, but they arrive at the I/O buffering queue at random times $\tau_1, \tau_2, \ldots$, and the Dma data that arrives at time τ_i requires X_i pclocks (discrete units) of time that it stays in one of the 3 buffers before it can be reused. Based on the following idea(Davis 1993) the total service as a function of the pclocks at discrete time t is

$$S_t = \sum_i X_i I_{(t \geq T_i)}, \tag{8.9.1}$$

assuming that there is zero service load existing initially, $I_{(.)}$ is an indicator function which is unity when the inequality in bracket (.) is satisfied and zero, otherwise. In addition, the so–called *virtual waiting time* W_t is the unique solution

[2]We wanted to call it Davis buffer.

to the equation

$$W_t = S_t - \sum_{j=1}^{t} I_{(W_j>0)}. \tag{8.9.2}$$

W_t is the amount of pclock time units which a DMA request data arriving at time t will have to wait before entering the I/O buffering system.

We decided to treat the lifetimes of a GX write buffers as a service process and the problem now becomes a finite buffer queue with queue length 3. Using the $M/G/1/N$ transient queue model which we developed in the previous chapters. We developed a recursive algorithm that computes the queue length over a finite time period. The results of the queue length (No. of cache lines in the I/O) as a function of time (in processor clocks– pclks) are shown in 8.10, 8.11, 8.12 and 8.13. The model was also able to predict a linear relationship between the lifetime of a GXC buffer and the number of cache lines in the buffering system for a given I/O port BW (see figure8.14). This suggests an operational condition that restricts us to stay within 2.0 caches as much as possible, so that any arbitrary Dma Write that arrives is not blocked due to all the 3 buffers being full.

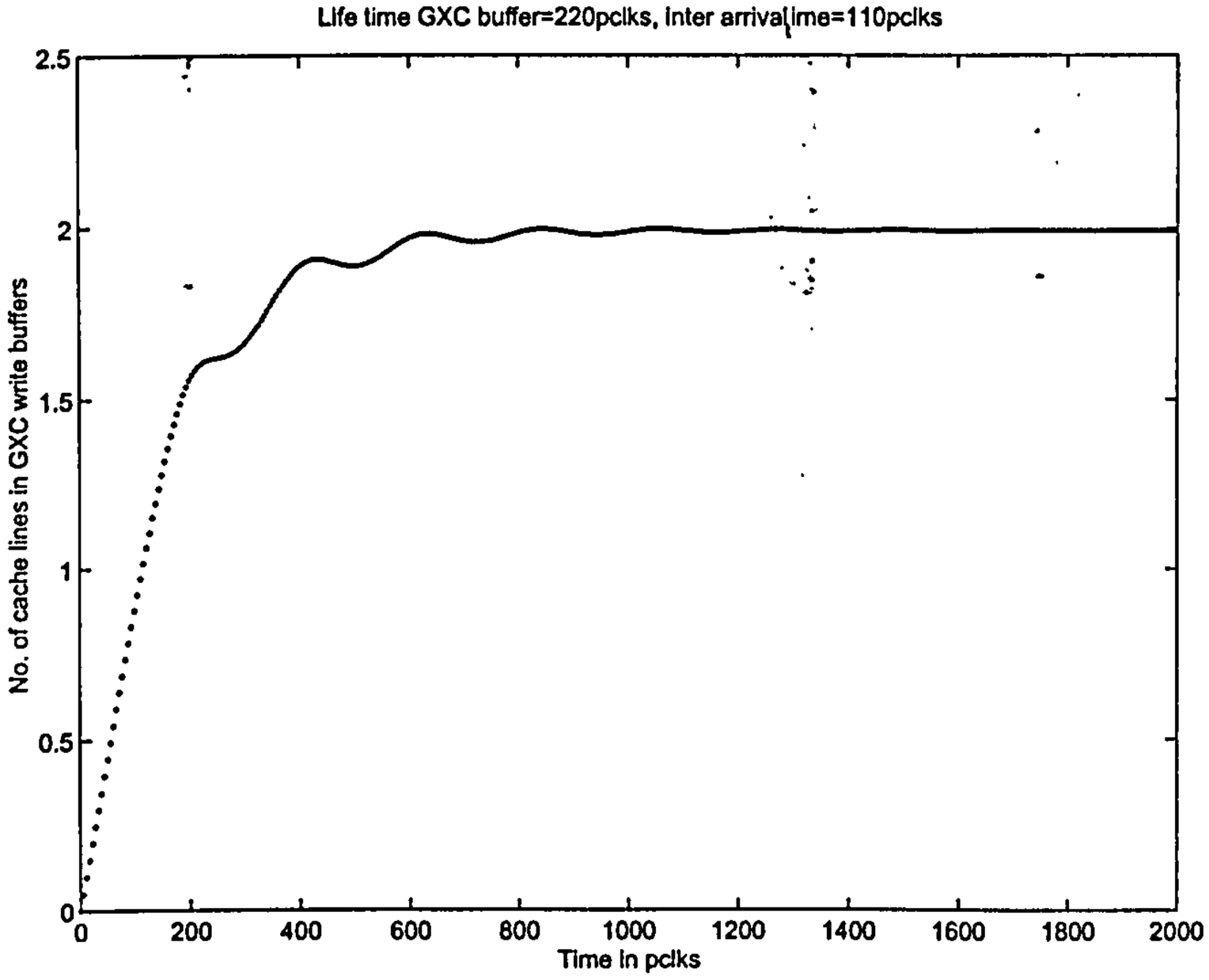


Figure 8.10: Queue length with bandwidth 1.5GB/sec

That said, multiple adaptors are to be connected onto the SCI ring within which connects I/O and CPU units into the memory sub system. One way of approaching the multiple adaptors problem is to study the traffic paths across the

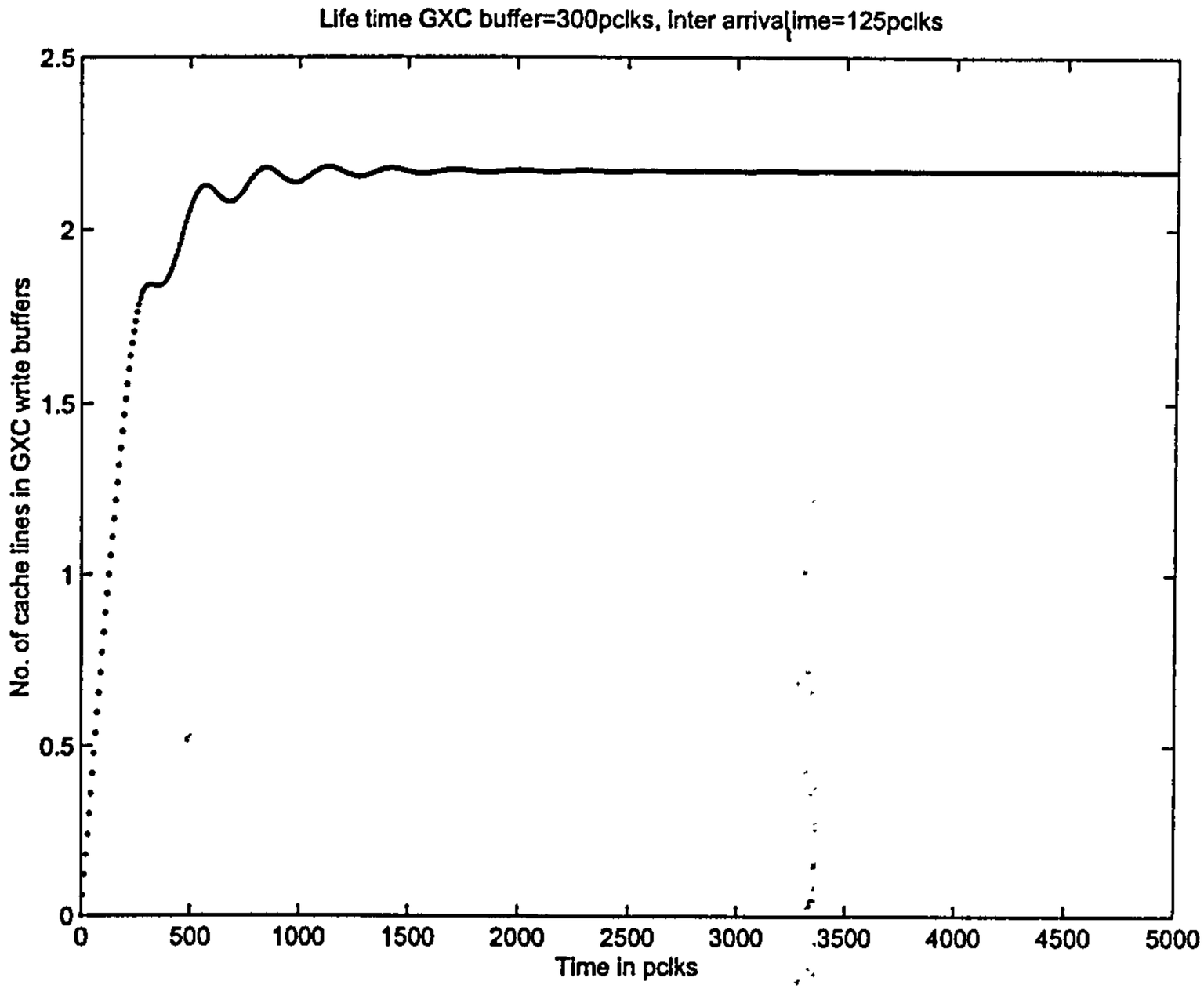


Figure 8.11: Queue length with bandwidth 1.3GB/sec and buffer time 300 pclks

inter–connecting fabric paths within the node and seek for the effective utilisation of the fabric busses. In this way if the effective utilisation turns out to be too large then there will be some back pressure onto the I/O busses that will reduce the aggregate multiple adaptors bandwidth. There is relationship between the ring utilisation and the bandwidth of the I/O bus which has been established through a queue simulation of the ring and also through a VHDL simulation.

The only effect that a high SCI ring utilisation has on the I/O system is to increase the lifetime of Write buffer. It should be observed that a relationship between the GX BW and the lifetime of a Write Buffer will provide a solution to the multiple adaptors problem. But we are able to get a relationship between the lifetime of the GX buffer and the SCI ring utilisation from a low level and queue level simulation of the whole computer system. Thus, the bandwidth problem is solved–the theory of PDP has provided us with the means of analysing and optimising that I/O bandwidth of a major message passing project.

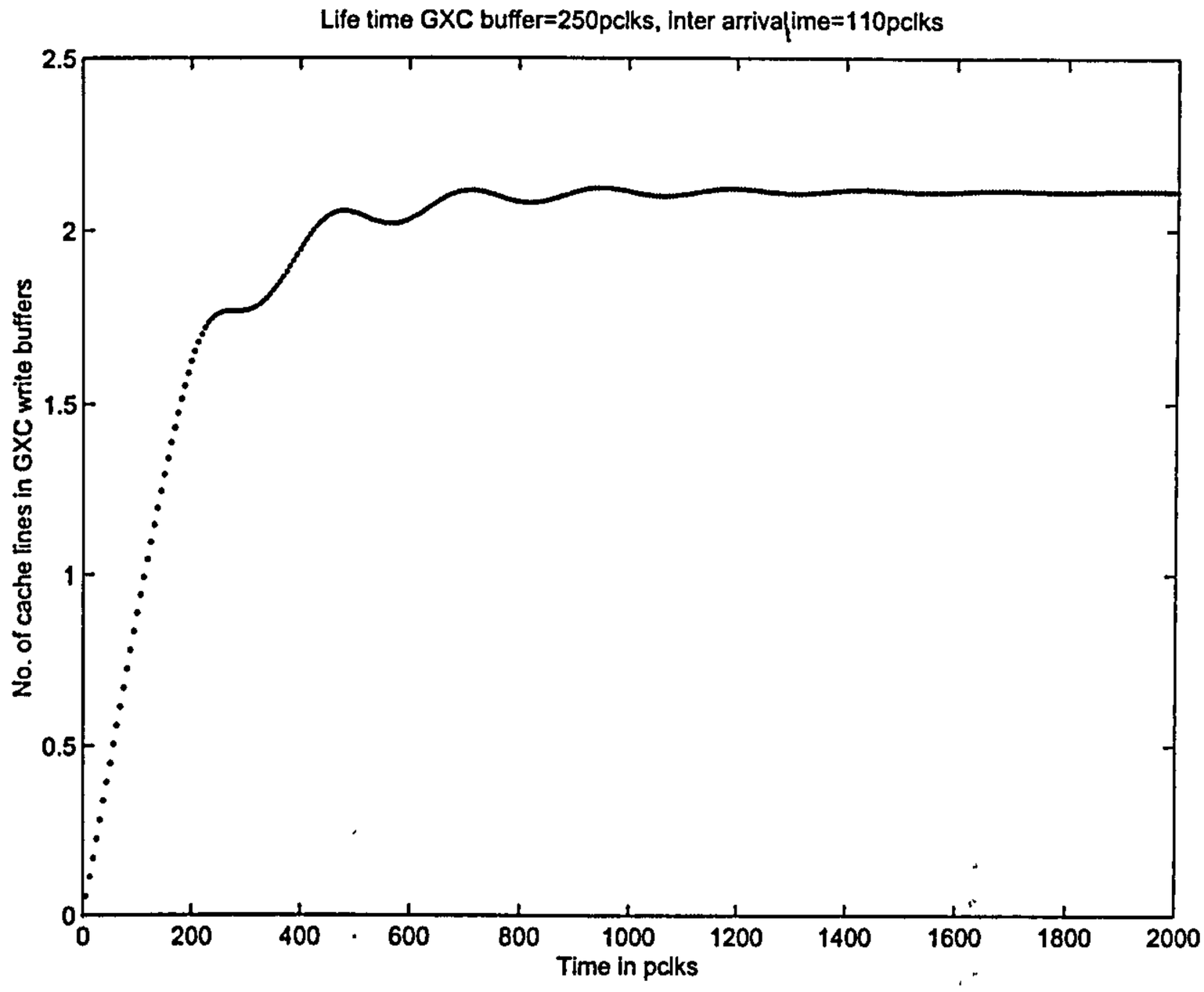


Figure 8.12: Queue length with bandwidth 1.5GB/sec and buffer time 250 pclks

8.10 Conclusion

In this chapter, PDP is applied for the first time to the modelling of data communication through a high bandwidth switch. This is an IBM switch that connects a number servers for a parallel processing application. We present a new approach to the problems of estimating and optimising bandwidth of data communication across the switch. The new PDP model is used to estimate the queue length of packets in the software protocol layer of the data communication. In addition, a similar framework is used to optimise the bandwidth of a hardware I/O component of this message passing operation between two servers that are connected to an IBM switch. Although we did not directly seek for a grand approach to modelling communication with PDP, the framework applied to the two examples discussed is hardly specific to them. Its principles are quite general and can be applied to other physical communication systems. In spite of numerous references to recent research for message passing across switch, genuine examples of transient queue models are missing. Therefore, suitable " stock examples" on which theory can be tested are not readily available. Therefore, many tools developed and applied here are comparatively new and unfamiliar. It is hope that they will(like the extension of PDP models developed in the previous chapters) be of interest to a wide class of researchers on system development.

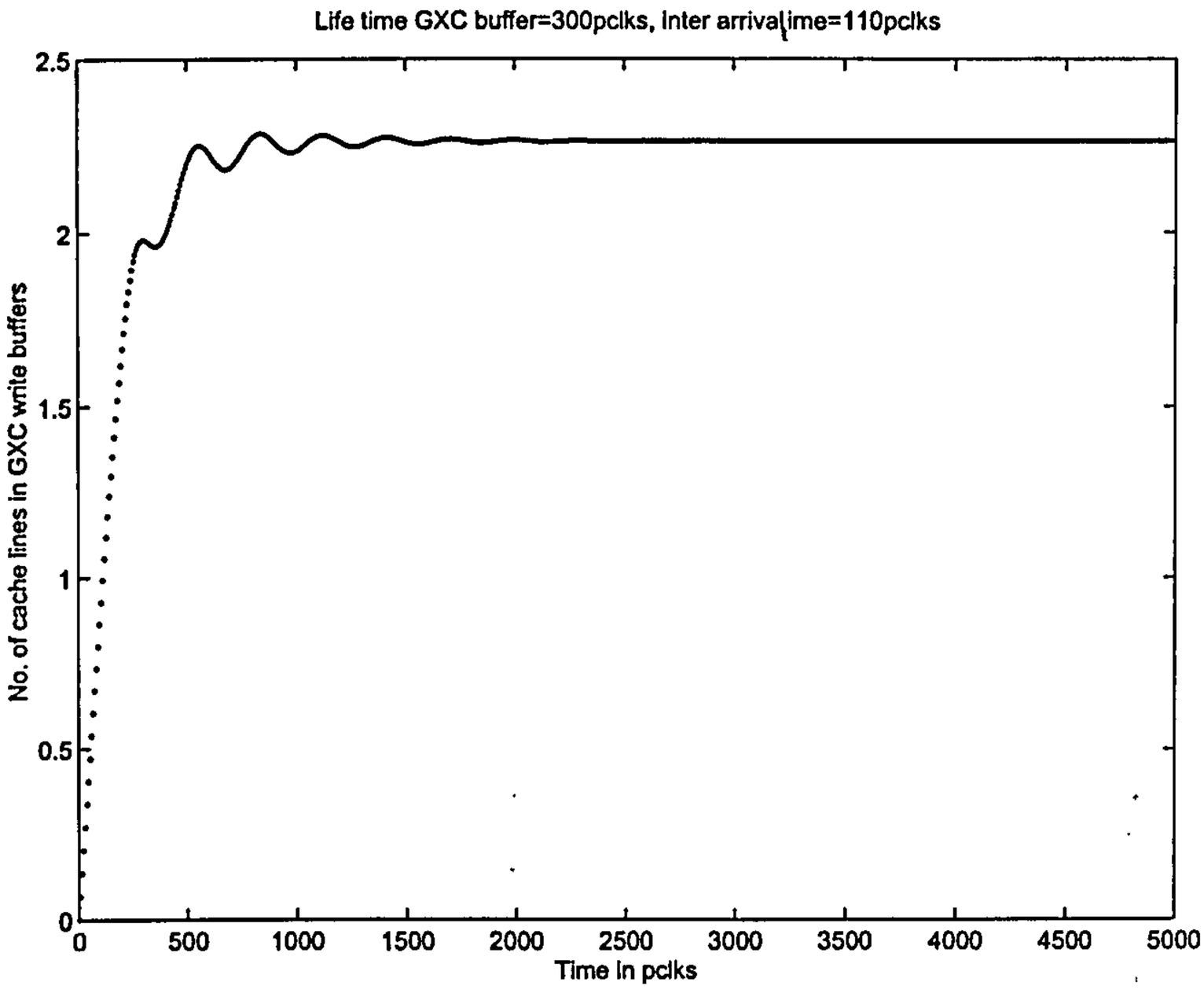


Figure 8.13: Queue length with bandwidth 1.5GB/sec and buffer time 300 pclks

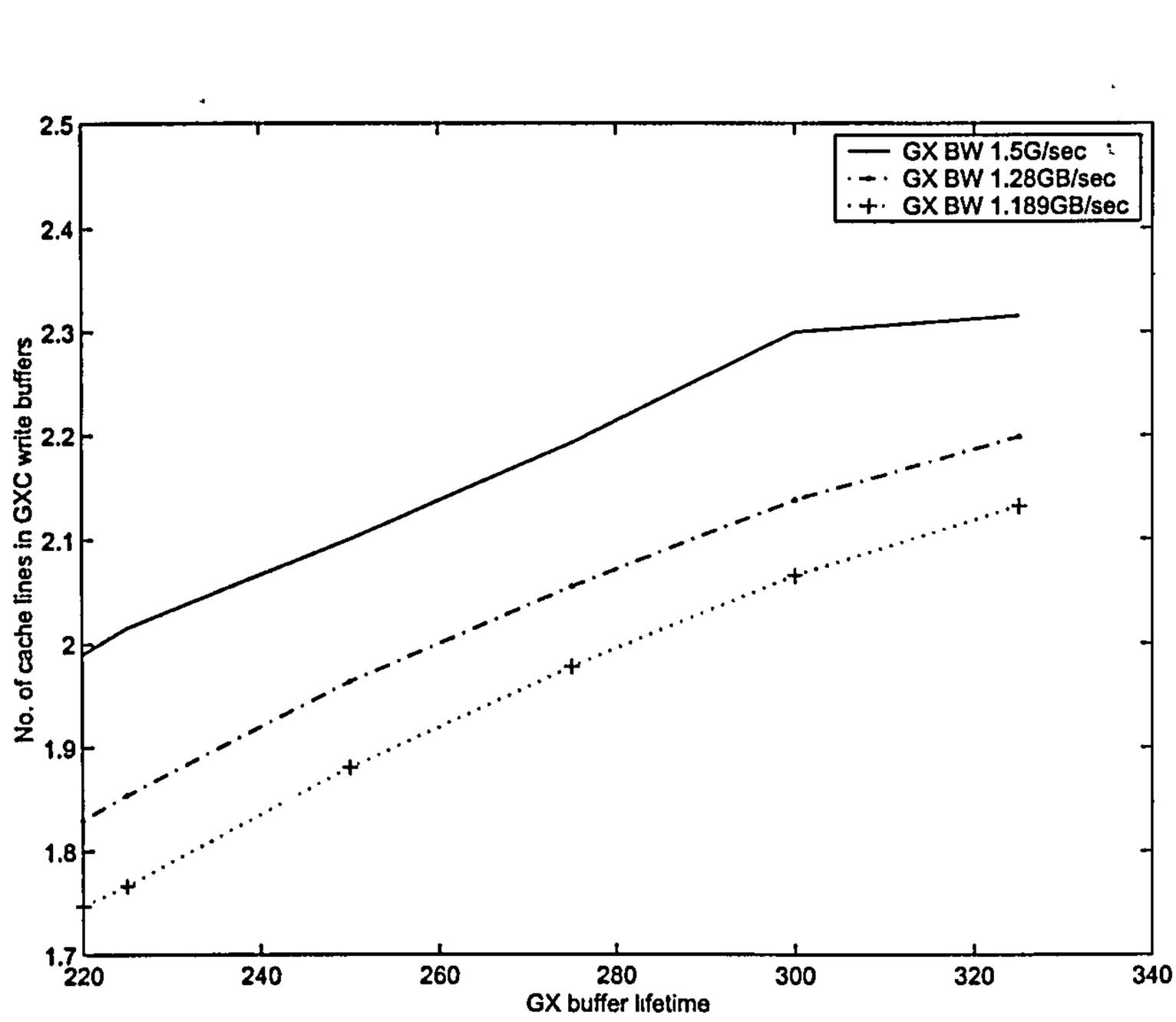


Figure 8.14: Queue length against buffering delay and bandwidth

Chapter 9

Conclusion

We set out in the thesis with a common goal: *a thoughtful insight might mitigate the difficulty of prediction demand levels arising from a compound renewal process.*

Using standard multi–layer neural networks to approximate these data sets, we have found it to be problematic in a number of ways, not the least of which is the poor convergence of network weights under training due to the inherent nonlinear nature of the weight updating algorithms. That is why our , attention has moved towards radial basis function networks (RBFs) which have much better adaptive properties.

This is mainly because of the straight forward linear relationship between the network weights and the output such that a linear parameter estimation technique can be employed for weight updating. RBF networks have the ability to model any arbitrary nonlinear mapping, even though the output layer of an RBF network is merely a linear combination of hidden layer signals, there only being one hidden layer. RBF networks therefore offer considerable promise in terms of network stability and robustness proofs, largely because the network can easily be defined by means of an equation set.

As is the case with modelling methods in general, and particularly with neural networks such as MLPs, a problem occurs in terms of determining a suitable RBF network structure.The key problem is deciding both the quantity and positions of basis function centres , in that the choice directly affects the goodness of fit of the overall network approximation capabilities achieved, with the requirement that they must certainly be sufficient in number to span the entire input domain. However, the use of a large number of basis function centres obviously raises questions in terms of computational effort, particularly insofar as the number of

basis function centres increases exponentially with regard to input space dimension. Additionally, this can also create the danger of over–parameterisation leading to the loss of generalisation and in this way it can be considered that RBF networks become when the input space is of high dimension, or in other words RBF networks are best suited to less complex problem

Turning to our own contribution, we put our finger on the central problem, when in our introduction, we remarked that the sort of data sets that we have considered are marked contrast from those normally employed to the study of new algorithms in machine learning. Our data sets are derived from demand and supply processes that are very complex. In machine learning research literature, too much attention has been put to study of learning structures *per se* too little to the study of the performance of a specific learning structure and the efficiency under different noisy conditions or unusual stochastic regimes.

To assess the efficiencies, limitations and relative merits of algorithms,examples were chosen carefully to give a fair picture. In our first approach, we consider data sets that demonstrates the *strongest* differences from those usually employed to illustrate new algorithms in machine learning.

The attempt in this work is to provides a simple framework of building learning system which is capable of learning and analysis simultaneously. The choice of our learning algorithms have been justified and the optimisation routines used are carefully outlined and , where appropriate, mathematical explanations are given. Keeping in mind that the burning problem is concerned with certain pitfalls associated with the design of learning systems, we have concentrated on analysis of algorithms, subjecting them with data sets that are in very different from those normally used. We have also pointed out, where necessary, the theoretical knowledge to support our analysis. In addition, each chapter is punctuated with simulation results from our models.

9.1 Achievements

The major problem is to predict demand workload from a process consisting of a sum of individual processes. We framed that this process is a compound renewal process and adopted a Markovian framework for prediction. We developed and applied PDP Markov renewal theory which provides powerful tools for analysis. Thus, our contributions can be summarised as follows:

- We have introduced various Gaussian approximation of renewal processes and the application Montroll–Weiss equation to analysis of demand prediction in compound renewal processes. This provides us with techniques to deduce some Gaussian limits theorems.

- As we have explained in chapter (2), model reduction in data communication and RBFs regression has been formulated for controlling the complexity of learning structures and training data.

- New optimisation algorithms were developed and tested and method of analysis of variance is implemented to solve the model reduction problem. Results are used to evaluate data communication design and queueing simulation.

- Optimisation algorithms studied and reported in (Tankeh, Azvine and Mamdani 1997a, Tankeh and Mamdani 1997d) are extended and applied to real industrial problems.

- New methods for studying the relative merits among non–linear Markov models are presented. The analysis provides means for comparing Neural networks and RBFs. We developed a systematic method for comparing the relative merits of models, a technique based on the analysis of variance that goes beyond standard practice in RBFs research.

- Most works in data communication and RBFs assume that the noise is not correlated. Here we argue that this is not true for most compound renewal processes. And we show to do data analysis with RBFs and regressive data when the noise is correlated and the variance is not stationary. In machine learning research, it is normally assumed that the noisy terms that enters the learning structure are uncorrelated. It is too much to expect most random variable that describe an industrial process to be all independent. .

- The study of positive definite matrices has undoubtedly been going on since matrices were first developed. However, recent development under SDP provides new impetus towards the optimisation of covariance constraint problems, opening up new opportunities and greater awareness. This will in the future lead to widespread appearance and utility of SDP. This thesis has highlighted some of the many ways in which SDP arise in model and data reduction. It has illustrated how the properties of positive definiteness facilitate computation of reduction schemes, factor analysis and optimal design of training data sets.

- We have presented new methods for explicit analysis of a finite–buffer $M/G/1N$ queueing system. Firstly, we formulate the problem as a piecewise deterministic Markov process (PDP), relating the performance parameter of interest as a solution of a system of first partial differential equations (PDEs) of the hyperbolic type. Secondly, by setting up variables and the boundary conditions of the system of PDEs appropriately, we were able to have the equations for the transient queue length.

- We have developed two methods of solving the partial differential equations resulting from queues. The first is an implicit algorithm which is developed from on a finite difference scheme. A second method, which is more applicable to the message passing of chapter (8), is developed using an *upwind scheme.*

- This numerical results shed much light on the potential of PDP modelling and provides the first entry to the understanding of integro–differential which, hitherto, has barred real applications of PDP to the important problems listed in (Davis 1984).

- We have developed a complete new PDP discrete model and a discrete renewal theorem is proved for the first time using PDP theory.

- In our case–study work we model a unidirectional MPI set up in which messages flow from a sender node to a receiver node. We were able to predict in transient time the queue length, representing the number packets that are still in the pipeline in the computer memory that has not been acknowledged or transmitted.

9.2 Future directions

- The application of Montroll–Weiss equation to compound renewal process (Montrol and Weiss 1965) needs to be generalised and extended to more Gaussian time series and queueing problems.

- We have introduced the idea of extracting the mean of queue length from outputs of simulation. This idea can be explored for general stochastic time series with correlation.

- We solved the two-servers problem in the transient form and an extension to multivariable server model is needed.

- We expect to see the method we have developed for $M/G/1/N$ and $M/G/2/N$ to be extended to networks of queues.

- More work needs to be done to handle integro–differential equations arising from the generators of PDPs. One such line of work is in the area of functional equations and difference–differential equations, power series solutions and the use of complex variable.

- The application of $M/G/1/N$ to data communication in MPI and parallel file systems I/O have been demonstrated.

- The application of $M/G/1/N$ to *zero copy* data communication has been developed.

Chapter 10

Appendix: Data sets

10.1 Introduction

Data mining isn't likely to be fruitful unless the data you want to use fulfills certain criteria. These are some of the criteria considered:

1. **Data Availability**
   This may seem obvious, but be aware that although data might be available, it may not be in a form in which it can easily be used. The data, however, might be held in some other form, on a machine which can't be directly accessed. It will need to be down-loaded or "dumped" in a suitable form before it can be used. It might be scattered among different databases and sources, and needs to be pulled together. It may not even be on-line; perhaps it exists only on paper, requiring a data entry phase before data mining can begin.

2. **Does data cover the relevant factors**
   The object of data mining is to identify relevant factors, so this may seem like an odd question to ask. It is very useful, however, to look at what data is available and try to identify likely relevant factors which are not recorded. In trying to predict electricity consumption, for example, you may have lots of information on customers, consumption histories, etc, but you may not have weather and temperature information which is likely to play a significant role.

   Missing factors don't necessarily mean that data mining will not produce useful results, but they can limit the accuracy of predictions achieved.

3. **Is the data noisy?**

Data often contains errors, or may contain subjective and therefore variable judgments. The phenomena are collectively referred to as "noise".
Sometimes, noise in data is quite normal; there may well be underlying rules, but they may not hold for 100% of cases.

Typically, the more noise there is in data, the more difficult it will be to get accurate results.

4. **Is there enough data?**

This is a difficult question to answer, in data mining, it isn't the size of data set which is important. The "representativeness" of the data set is far more significant, together with its coverage of possible outcomes and combinations of variables. Typically, the more attributes that are considered, the more records will be needed to give representative coverage. If the data is representative, and there are general underlying rules, it may well be that a data sample of a few thousand (or even a few hundred) records will give equally good results as a million, and a lot more quickly.

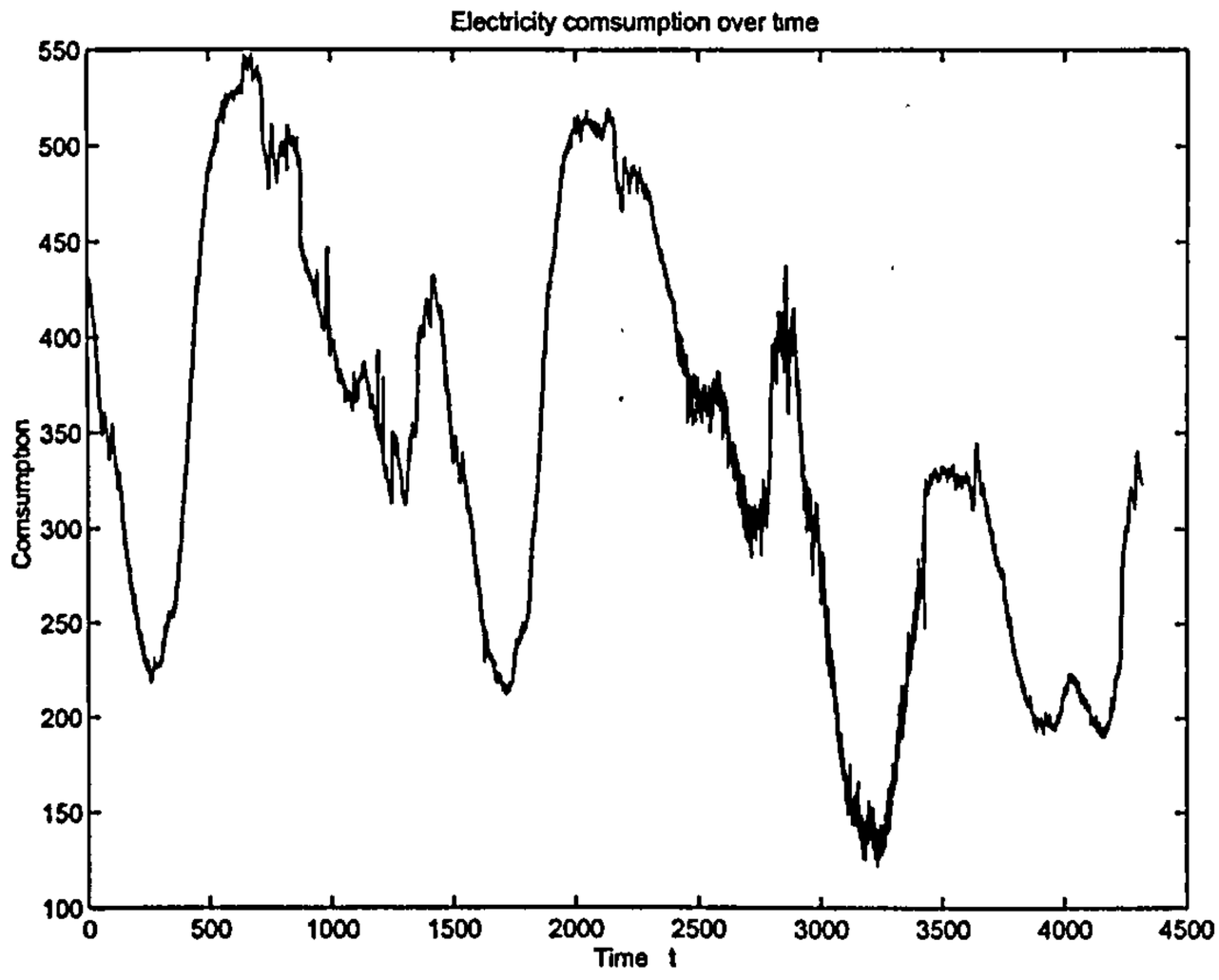


Figure 10.1: Electricity consumption

10.1.1 Electricity consumption data sets

Look at figure (10.1) which depicts electricity consumption over a period of about three days, sampled minute by minute. This represents a 4,500 point time series. Consumption has been measured in electricity units and time in minutes. Because

the load consumption curve is corrupted by noise, attempts to extract a signal from it will need to address the following questions:

1. What information is contained in the signal, and what pieces of information are useful?
2. Are there various kinds of noises, and can they be distinguished from one another?
3. Are there any seasonal components?
4. Is there any trend?
5. Are there any repetitive patterns, hence meaningful pattern?

As regards the data used in figure 1, the following points hold:

- The load curve is the aggregation of hundreds of sensors measurements, thus generating measurement errors.
- Roughly speaking, the consumption is accounted for by industry for 50% and by individual consumers for the other half.
- There are more than 10 millions individual consumers.
- The fundamental periods are the weekly-daily cycles, linked to economic rhythms.
- Daily consumption patterns also change according to rate changes at different times e.g. relay-switched water heaters to benefit from special night rates).
- Missing data have been replaced.
- Outliers have been corrected.

10.2 Insurance–claims

The following tables as shown in table exhibit that insurance claims is made on a daily basis for a 7 day week over a seven year period. A daily sampling basis, therefore, makes for a 2556 points time series.Figure (10.2) depicts actual daily claims plotted for the mentioned point time series. Insurance claims are measured

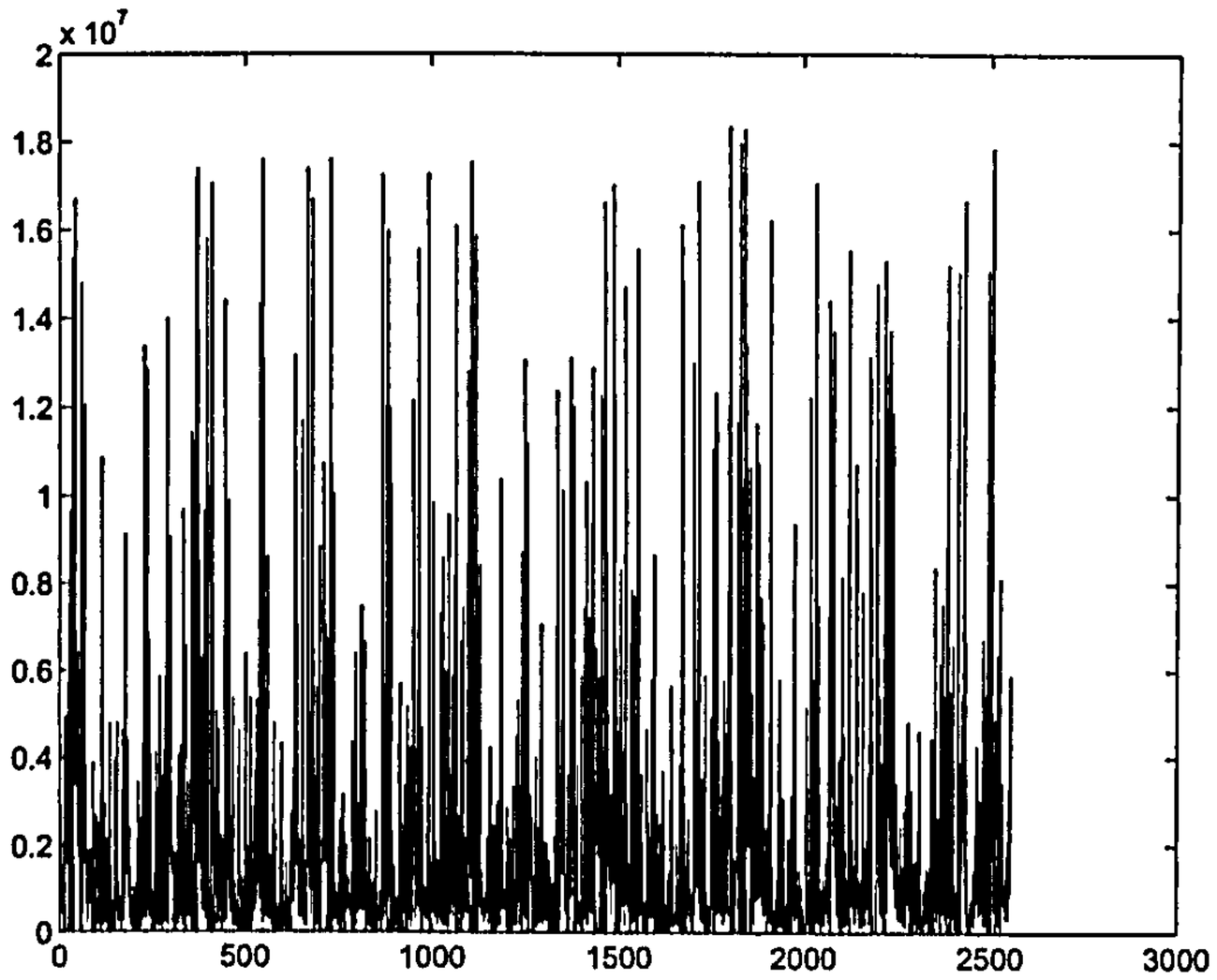


Figure 10.2: Daily claims over 7 years

in currency units and time, t, in days. Unlike in the electricity consumption case where the number of people turning on their light is not given, the number of individuals making insurance claims per day is and forms a count data set which has been plotted over the 2556 point time series. See figure (10.3).

In relation to the data used, the following points are worthy of mention.

- Missing data have been replaced.
- Outliers have been corrected, in this case 5% of the highest and lowest observations have been excluded.

10.3 Preference scores: tea data set

When designing new food products, companies will typically obtain data from three sources: sensory

panels, preference panels, and instrumental data. The nature of instrumental data is product-specific, but may include digital images, acoustic imaging or chemical fingerprinting.

The sensory panel[1] is a group of typically 10-20 people, selected and trained

[1]The preference and sensory panel data may be good from UCL Faraday project home

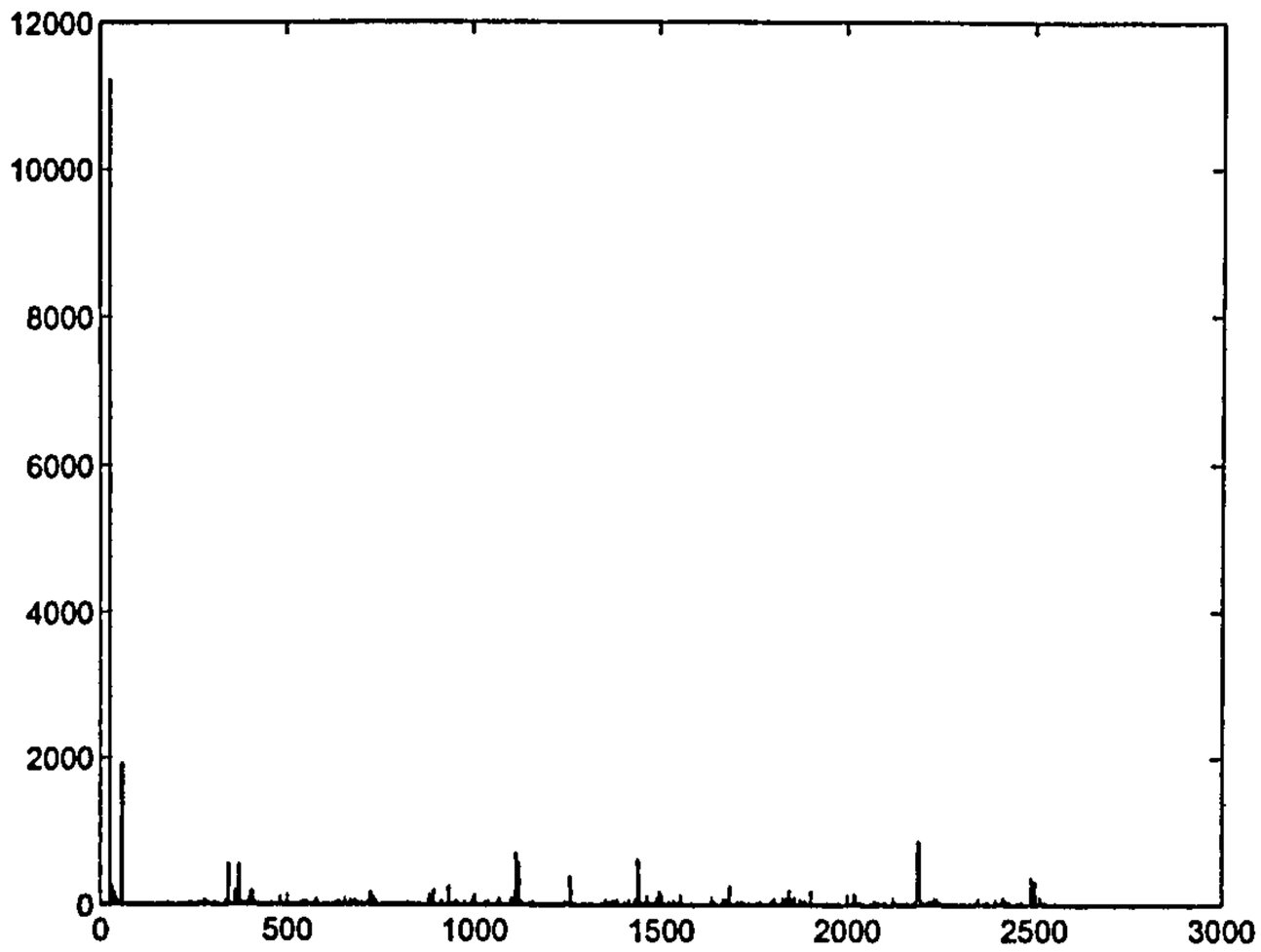


Figure 10.3: Daily claims over 7 years

for several months. The panel derives their own descriptors of product attributes which can then be systematically used to describe different varieties of the product. The panel typically produces between 8 and 20 descriptors after discussion and analysis. Members of the panel are then presented with a variety of different products, selected to represent a wide range of flavours, colours, etc.

They then measure each sample by ranking it for each descriptor. The ideal sensory panel

should produce absolutely consistent and uniform results, allowing the panel to be

treated as an instrument. In practice, human perception is neither absolute nor constant.

The preference panel is a larger group of untrained people, typically 50-500 potential

consumers, who are bought in "off the street" specifically for the trials. They are individually

presented with a few samples and are then asked to rank each one on a simple preference scale.

No training is given and no discussion between panelists is allowed, so the results will

---

page.

be entirely subjective, and vary from panelist to panelist. The relatively large panel

size should smooth out any unwanted discrepancies.

Once the preference panel has classified the samples, the sensory panel data is re–examined, to determine which sensory attributes best distinguish the different

preferences. For example, suppose the preference panel gave two samples significantly

different grades. Then if the sensory panel gave both of them the same grade for some

measure, e.g. shape, then this attribute is a poor predictor of quality. If a correlation

can be found between the one or more of the sensory panel attributes and the

preference panel scores, then this can be used to guide future product design and marketing.

In typical studies, each consumer considers each product, four or five at a sitting, over a total of two days. Any more testing might lead to taste-fatigue and so limits the total number

of products in studies. The first sample given is a dummy -

panelists tend to give disproportionately high scores to first sample

tasted (or viewed). One sample is repeated (unknown to the panelists)

to allow consistency checks within each panelist's scoring.

The entire data-gathering process is very expensive and very time-consuming,

and depends on human perception, which lead to the most striking and important features of the data sets: they are small and sparse, and contain uncertainty.

Unilever recently carried out a study in Russia, where tea purchase decisions are based primarily on the appearance of the leaves. The data covers 20 samples of tea. For each sample, the sensory panel gave measurements using 8 attributes and a preference panel of 450 people gave a single preference score.

Along with the panel results, digital images of all the samples were captured, and 9 features were extracted, measuring leaf size, texture and uniformity

This work was done externally by researchers at Brunel University.

Unilever is currently extracted a similar range of

features from the 20 tea samples provided by Unilever, and a further 17 samples purchased

on the High Street. No preference or sensory panel data is available for these samples,

but it is hoped that by combining this data with the labeled data (labeled with

preference scores and/or sensory scores), better results can be achieved.

Each row represents a single sample of tea. Column P is the mean preference,

and $S1 - -S8$ are the eight sensory panel scores.

10.4 Education test scores

The sample consisted of year–10 pupils of age 15 from Woodside Community School . The pupils have studied mathematics in the same school in mixed ability groupings in their first two years of secondary education and are streamed when entering year 9 . All of them are in the top set in mathematics . Before my teaching practice they have never been taught algebra in any serious way and it is customary of the Department to give training in algebra to year 10 . So the timing of my teaching practice coincided with the course in algebra . I was happy to seize the opportunity . I subjected them to an intensive tuition in algebra with considerable emphasis on formal operations on rules , logical explanations , demonstrations of understandings and problem solving .

The domain of algebra was wide , rich , saturated in contents and the learning atmosphere lively . As each topic was taught exercises were given to support learning , diagnose knowledge states and faults and for the purpose of record keeping and individual assessment . Each work sheet was graded up to 5 levels (A, B , C , D , E) . E and sometimes D were extended work on the topic demanding a considerable rich knowledge of the topic and its abstract relations to

some other branches of mathematics .

The test I gave was a 32–item diagnostic test designed by Gutvirt and administered by Birenbaum (1) . This test is on the solution of linear equations in one unknown . It is given in Appendix A . More details of the test statistical parameters are given in the references mentioned in chapter (4).Also we gave a profile of the test and the attributes to describe the possible operations in the solutions of the linear equations and again I learnt this from the works of Birenbaum.

10.4.1 Production yield over different batches

The for the data of the production plant. As shown the points fall mainly in the northwest and southeast corners of the diagram, bringing out the negative dependent between neighbouring batches evident in figure (3.14). The graph is shown in Chapter (3).

10.5 Product sale prices

This is a daily sale price of a product that is recorded over 2001 days. The data are highly correlated and volatile.

10.6 House valuations in different neighbourhoods

Data were collected on five variables $(x_1, x_1, x_2, x_3, x_4, x_5)$ on each house that were going on sale in different places. This example is the extension on the tea preference analysis[2] and we have replaced the 9×20 data with a vector of five variables representing;

- A letter form A, B, C, D, E, F, G.
- x_1 = Appraised land value.

[2]We actually used factor analysis here but in this part of the investigation we are restricted by Unilever from publishing the results of the total length of the tea data of dimension $9 \times 20 \times 500$, although we have been permitted to use the mean average summary of limited dimension 9×20 which we have used throughout out this work

- x_2 = Appraised value of improvement on the home.
- x_3 = Total appraised value of the home.
- x_4 = Sale price.

The test scores of house sales were conducted in 7 different neighborhoods A, B, C, D, E, F, G. The dimensions of houses examined in the seven neighborhoods were $(42, 319, 258, 467, 500, 372, 482)$, a total of 2440 data points, i.e. 42 houses were examined in A, 319 examined in B, and so on.

10.7 Rubber consumption

To illustrate the kind of data reduction and model reduction methods we are studying, a small test sample from the data mining inventory at UCL is used. The data set has four attributes and one response.

Response	Attributes			
y	x_1	x_2	x_3	x_4
0.909	1.287	0.984	0.987	1.046
1.252	1.281	1.078	1.064	1.081
0.947	0.787	1.061	1.007	1.051
1.022	0.796	1.013	1.012	1.046
...	...	...	...	...

Only the first four data entries are given above. The full dataset consists of sixteen such entries describing the total annual demand of rubber from 1948 to 1963 along with attributes such as car production, gross national product, etc., which we might expect to affect the annual demand for rubber, although we cannot be certain of this and, more importantly, we do not know whether they are sources of independent information. It is a small example for illustrative purposes only but, within the context of this Faraday project it could be regarded as similar to the sort of data produced in a food design study, although it is probably less variable.

10.8 Basic fact about testing F–hypothesis

A very commnon feature of statistics is a procedure known as hypothesis testing for determining if an assertion about a characteristic of a population is reasonable. In many instances we formulate a statistical for the sole purpose of rejecting or nullifying it. Similarly, if we want to decide whether one procedure is better than another, we formulate a hypothesis that there is *no difference between the procedures* (i.e. any observed differences are merely due to fluctuations in the sampling from the same population). Such hypotheses are often called *null hypothesis* and are denoted by H_0. An alternative to the null hypothesis is denoted by H_1. With respect to data analysis, there are three distributions that are very important to statistical test: the chi–square distribution , the t– distribution and the F–distribution:

1. Let $x_1, x_2, \ldots, x_\nu$ be ν independent normally distributed random variables with mean zero and variance 1. Now consider the random variable

$$\chi^2 = x_1^2 + x_2^2 + \ldots + x_\nu^2 \tag{10.8.1}$$

the distribution define by

$$F(x) = P(\chi^2 \leq x)$$

is called chi–square with ν degree of freedom. The χ^2 distribution is skewed to the right. The plots in figure (10.4) are those of this distribution with five degrees of freedom and seven degrees of freedom.

2. Suppose that y and z are random variables, where y is normally distributed with mean 0 and variance 1 while z is chi–square distributed with ν degrees of freedom. Then the random variable

$$T = \frac{y}{\sqrt{z/\nu}}$$

has t distribution with ν degrees of freedom. In figure (10.5) the plot compares the t–distribution with $\nu = 4$ to the shorter tailed standard normal distribution. If ν is large ($\nu \geq 30$) graph of the density approximates the standard normal.

3. Let x_1 and x_2 are independent random variables which are chi–square distributed with ν_1 and ν_2 degrees of freedom, respectively. Then the random

variable

$$V = \frac{x_1/\nu_1}{x_2/\nu_2}$$

has the F distribution with ν_1 and ν_2 degrees of freedom.

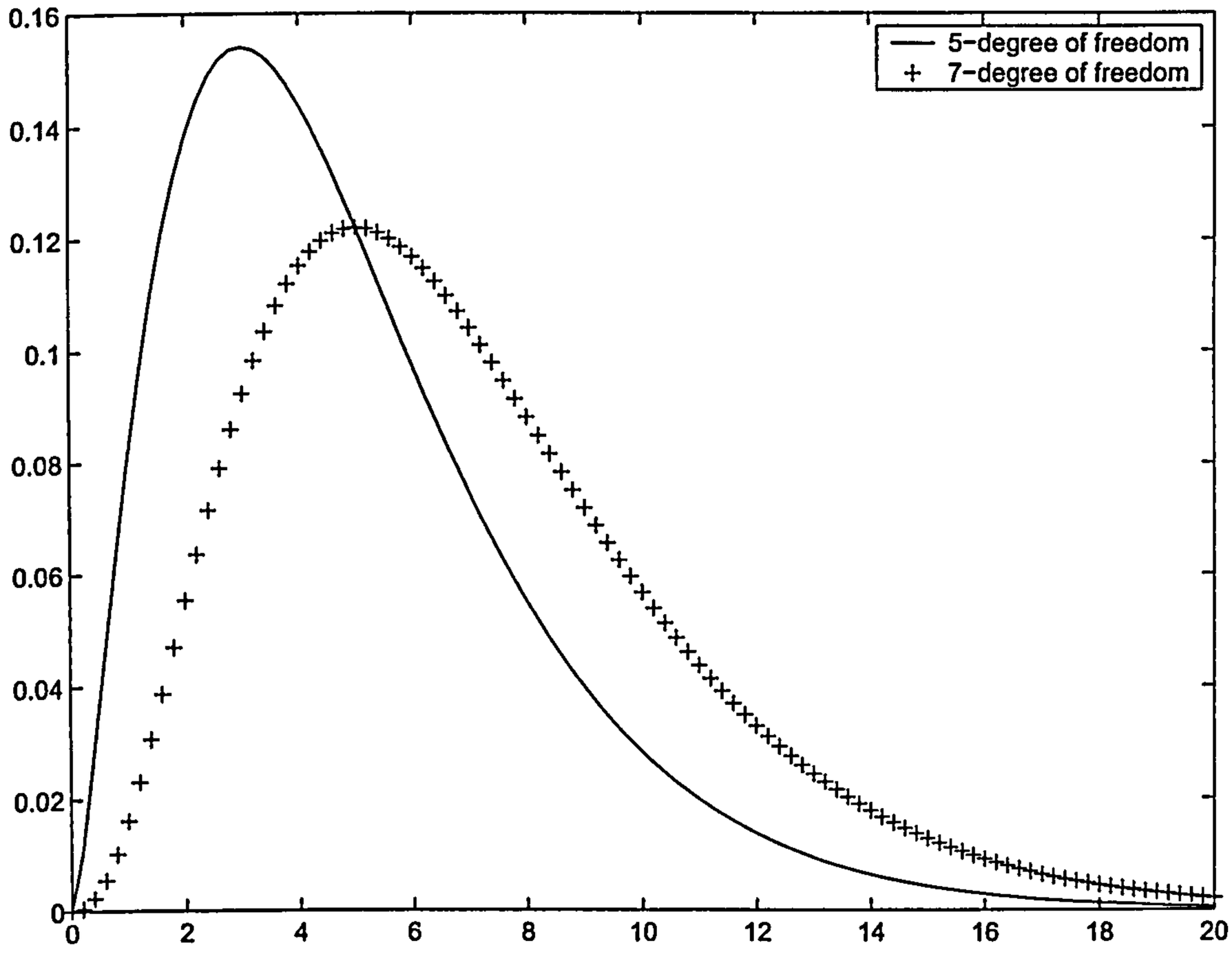


Figure 10.4: Chi–square distribution

10.8.1 Covariance and schur complements

A good point of contact between SDP and data mining is the covariant matrix $\sum$ of a multivariate $z \in R^n$. It is well known that $\sum$ is symmetric and positive definite. Hence, it belongs[3] $S^n_{\succ}$. It follows that the density $f(z)$ of z is given by

$$f(z) = \left(\frac{\det \sum^{-1}}{(2\pi)^n}\right)^{1/2} \exp\left\{-\frac{1}{2}(z-\bar{z})^T \sum^{-1} (z-\bar{z})\right\}, \qquad (10.8.2)$$

where $\bar{z}$ is the mean of n–variate z, see (Davis and Vinter 1984).

Due to the positivity of $\sum$, all its principal sub matrices are all positive definite and invertible. Anticipating model/data reduction, as we shall see earlier, assume

[3]We assume to be working over the real field.

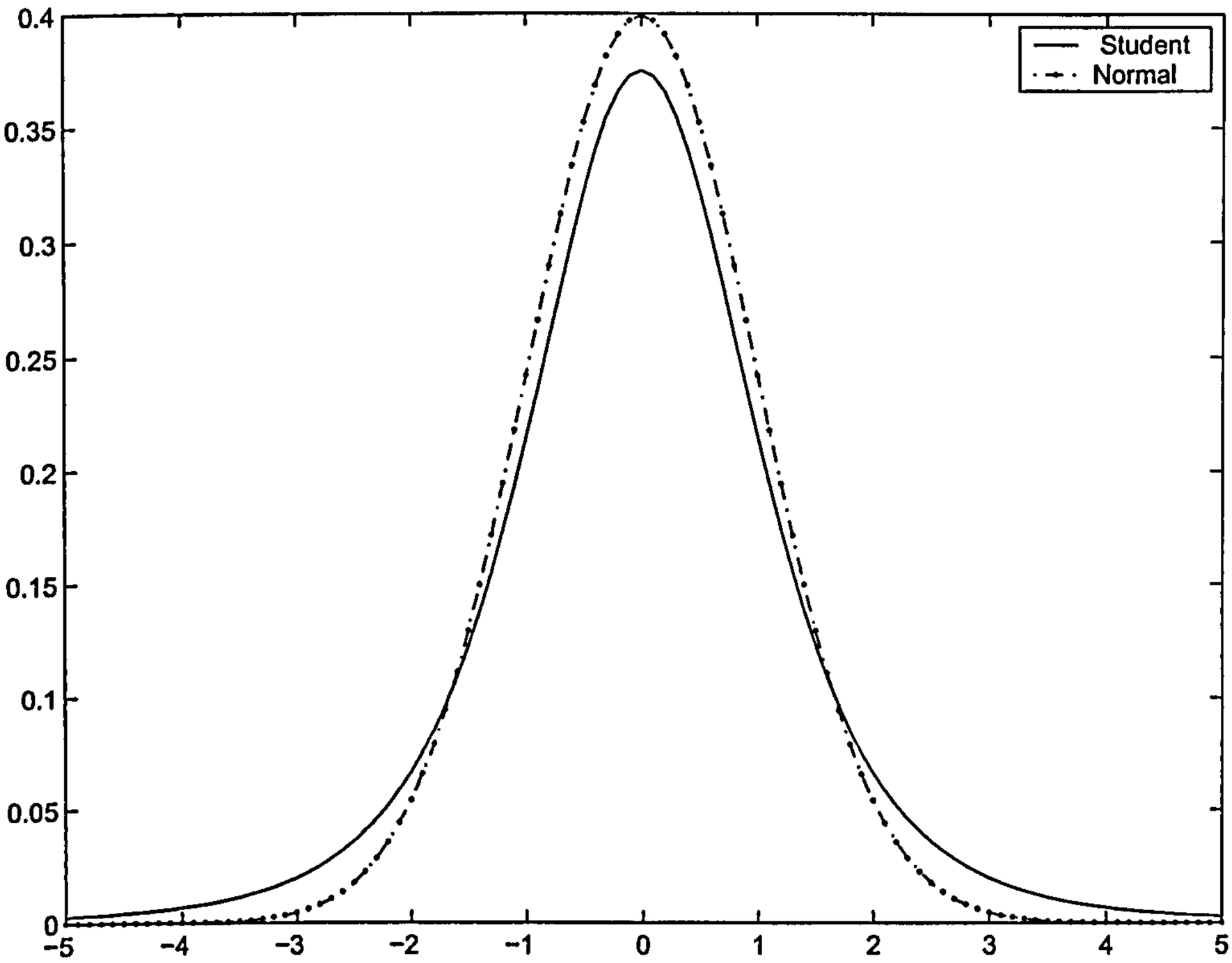


Figure 10.5: Student distribution

the following partition

$$z = \begin{pmatrix} x \\ y \end{pmatrix} \quad \text{and} \quad \bar{z} = \begin{pmatrix} \bar{x} \\ \bar{y} \end{pmatrix}, \tag{10.8.3}$$

where x and $\bar{x}$ belong to R^{n_1}, y and $\bar{y}$ belong to R^{n_2}, $n_1 + n_2 = n$. Additional advantages accrue from the fact that $\sum$ is symmetric[4] with the following partition

$$\sum = \begin{bmatrix} A & B \\ B^T & C \end{bmatrix}. \tag{10.8.4}$$

Associated with (10.8.4) is the quadratic form

$$Q(z) = Pax, y) = x^T Ax + 2x^T By + y^T Cy. \tag{10.8.5}$$

[4]This property can be used to reduce the number of indeterminancies in relationships in Frisch scheme as we shall see later.

The *conditional distribution of x given y (written as x/y)* has density

$$f(x/y) = \frac{f(x,y)}{f(y)}, \tag{10.8.6}$$

where $f(y)$ is the density for the *marginal distribution of y,* given by

$$f(y) = \left(\frac{\det C^{-1}}{(2\pi)^{n_2}}\right)^{1/2} \exp\left\{-\frac{1}{2}(y-\bar{y})^T C^{-1}(y-\bar{y})\right\}. \tag{10.8.7}$$

The " fictitious" n_1– dimensional variable x/y has mean

$$\bar{x} + BC^{-1}(y-\bar{y}), \tag{10.8.8}$$

and covariance matrix

$$\left(\sum /C\right) = A - BC^{-1}B^T. \tag{10.8.9}$$

In SDP theory the designation " *Schur complement* " has been appplied to the matrix of form $A - BC^{-1}B^T$. Then $A - BC^{-1}B^T$ is the *Schur complement*[5] of C in $\sum$ and is denoted by $(\sum /C)$. Later, we shall show how $(\sum /C)$ in (10.8.9) is applied to data reduction using a method based on principal component.

Returning to the quadratic form $P(x,y)$ in (10.8.5) we can express the gradient $g(x) = \min_x Pax, y)$ as

$$g(x) = x^T(A - BC^{-1}B^T)x = x^T\left(\sum /C\right)x. \tag{10.8.10}$$

[5]In memory of the mathematician Schur who first put it to use in 1917.

Chapter 11

References

Bibliography

[1] Ablowitz, M.J. and Fokas, A.S. (1997) Complex variables : introduction and applications.

[2] Allwright, J. (1988) Positive definite matrices: Characterization via conical hulls and least–squares solution of matrix equation, *SIAM J. Control optimisation,* Vol. 26, pp.537– 556.

[3] Allwright, J. (1989) On the maximizing the minimum eigen value of linear combination of symmetric matrices, *SIAM J. of Matrix Anal. and Appli.* Vol. 10 pp.347–382).

[4] Amari, S. (1989) Learning patterns and pattern sequences by self–organising nets of threshold elements. *IEEE Trans. Comp. 21,* 1197–1206.

[5] Andersen, E-Sparre, (1957) On the collective theory of risk in the case of contagion between the claims. Transactions: XV^{th} international congress of actuaries, New York. II 219–229,

[6] Anderson,T. W., Asymptotic theory for principal component analysis, *Annals of mathematical statistics,* Vol. 43, pp122–148.

[7] Anderson, B. D. O. and Moore, J. B. (1979) Optimal Filtering, *Prentice–Hall, Englewood Cliffs, NJ.*

[8] Apkarian, P. and Gahinet, P. (1995) A convex Characterisation of gain–Scheduled Controllers, to appear in *IEEE Trans. Aut. Contr.,*

[9] Araki, S., Bilas, A., Dubnicki, C., Konishi, K., Philbin, J., "User Space Communication - A quantitative Study", ACM and IEEE Computer Society Press, November 1998.

[10] Asmussen, S. (1987) Applied Probability and queues, *Wiley.*

[11] Atkinson A. C. and Donev A. N. (1992) Optimum Experimental Designs, *Oxford Statistical science series.*

[12] Barti, M. and Capuzzo–Dolcetta (1997) Optimal control and Viscosity solutions of HJB Equations, *Birkhauser.*

[13] Bardi M, Crandall M. G., Evans L. C., Soner H. M., Souganidis P. E (1995) Viscosity solutions and Applications, *Lecture notes in Mathematics–Springer.*

[14] Barnett, S. and Storey, C. (1970), Matrix methods in stability theory, *Nelson, London.*

[15] Beghelli, S. *et al* (1990) The Frisch scheme in dynamic system identification, *Automatica,* Vol.26.

[16] Bellman, R. and Cooke, K. (1962) difference– differential equations.

[17] Berenji, H., Chen, Y. Y., Lee, C. C., Jang, J.S. and Murugesan,S., (1990), A hierarchical approach to designing approximate- reasoning based controllers for dynamical physical systems. *In Poc. 6 Conf. on uncertainty in Artificial Intelligence* , (P. Bonisonne, Ed.), p.362,

[18] Bezdek, J. C. (1981) Pattern recognition with fuzzy objective function, *Plenum press, New York,* .

[19] Bhoedjang, R., Ruehl, T., Bal. H., "A Communication Substrate for Myrinet", *4th Conference of the Advanced School for Computing and Imaging,* Lommel, June 1998a.

[20] Bhoedjang, R., Ruehl, T., Bal. H., "User Level Network Interface Protocols", *IEEE Computer,* November 1998b.

[21] Billings, S. A., and Fadzil (1985), The practical identification of system with nonlinearities Proc. *IFAC Symposium on identification and System Parameter Estimation,* York, U.K pp.155- 160.

[22] Billings, S. A. Voon, W. S. F. (1986),Correlation based model validity tests for non-linear models. *International Journal of Control*, Vol.44, 235-244.

[23] Chen, S., and Billings, S. (1990), Practical identification of NARMAX model using radial basis function.*International Journal of Control* , 52, 1327-1350.

[24] Chen, S., Cowan, C. F. N. And Grant, P. M. (1991) Orthogonal least squares learning algorithm for radial basis function network. IEEE Trans. Neural Networks,2, 302-309.

[25] Chen, S., and Billings, S. A. (1992) Neural networks for non-linear dynamic modelling and identification. International Journal of Control , Vol.56, No.2, 319-346.

[26] Zhu, Q. M. and Billings, S. A. (1996) Fast orthogonal identification of nonlinear stochastic models and radial basis function neural networks, *International Journal of Control* , Vol.64, No. 5, 871–886.

[27] Zhu, Q. M. and Billings, S. A. (1996) Parameter estimation for stochastic nonlinear rational models , *International Journal of Control* , Vol.57, 309–333.

[28] Birenbaum . M , Kelly . A . E , Tatsuoka . K . K. (1993) Knowledge states in algebra, *International Journal of Research in mathematical Education* Vol.24 No 5, pg. 442–459

[29] Bischak, D. P., Kelton, W. D. and Pollock, S. M. (1993) Weighted Batch Means For Confidence Intervals In Steady-State Simulations. *Management Science, 39, 1002-1019.*

[30] Blackwell, D (1965) Discounted Dynamic Programming. *Ann. Math. Stats.* Vol. 36 pg. 226–235.

[31] Boel, R., Varaiya, R. and Wong, E. (1975) Martingales on Jump processes. *SIAM J. Control,* 13.

[32] Boyd, S., Ghaoui L. El., Feron E. and Balakrishnan V. (1994) Linear Matrix Inequality in Systems and Control, *SIAM Books, Philadelphia.*

[33] Bremaud, P. (1980) Point processes and queues: Martingale dynamics. *Springer–Verlag.*

[34] Broomhead, D. S., and Lowe, D. (1988) Multivariable functional interpolation and adaptive networks , *Complex Syst.* Vol.2. pp321-355

[35] Broerson, P. M. T., and Wensink, H. E. (1993) On Finite Sample Theory for Autoregressive Model Order Selection. *IEEE Transactions on Signal Processing, 41, 194-204.*

[36] Brown, M. And Harris C, (1994) Neuro-fuzzy Adaptive modelling and control, *Prentice Hall,.*

[37] Broyden, C. G. (1970) The convergence of a Class of Double–rank Minimisation Algorithms, *J. Inst. Maths. Applics.,* Vol. 6, pp. 76–90.

[38] Bulmann, H. (1970) Mathematical Methods in Risk Theory. *Springer–Verlag, .*

[39] Butcher, J. C. (1964) Implicit Runge–Kutta processes. *Math. Comput.* Vol. 18, pg 50–64.

[40] Butcher, J. C. (1987) The numerical analysis of ordinary differential Equations. *John Wiley, Chichester.*

[41] F. Chen. (1990) Back-Propagation Neural Networks for Nonlinear Self-Tuning Adapative Control, *IEEE Control Systems Magazine.*

[42] Chen, S. Billings, S. and And Grant, P. M. (1992) Recursive hybrid algorithm for nonlinear system identification using radial basis function network. *International Journal of Control* , 55, N0. 5 1051–1070.

[43] Chen, X. and Leslie I. M. (1992) Neural adaptive Congestion Control for Broadband ATM networks, *IEE Proceedings-I, Vol. 139, No. 3.*

[44] Cheng–Shang, C. (1995) Effective Bandwidth in High– Speed Digital Networks, *IEEE J.S.A. Communications,.*

[45] Choudhury, G. L. (Feb. 1996) Squeezing the most out of ATM, *IEEE Trans. on Communications,* Vol. 44, No.2.

[46] Chow, Y. S., Robbins, H. and Siegmund, D. (1991) The theory of optimal stopping, *Dover Books,.*

[47] Coppel W. A. (1965) Stability and asymptotic Behaviour of Differential equations, *Boston: D. C. Health.*

[48] Costas, O. L. V. and Davis, M.H.A. (1991) Impulse Control of Piecewise–Deterministic Processes Via Linear Programming, *IEEE Trans. On Auto. Control, Vol. 36, No. 3. 371–375. .*

[49] Costas, O. L. V. (1993) Discretizations for the average impulse control of PDPs, *Journal of applied probability,* Vol. 30, pg 405–420.

[50] Cox, D. (1962) Renewal Theory, *Methuen,* London.

[51] Cox, D. (1962) The analysis of non–Markovian stochastic processes by inclusion of summplementary variables, *Proc. Camb. Phil. Soc.* Vol. 51, pg. 433–441.

[52] Cox, D. (1962) Renewal Theory, *Methuen,* London.

[53] Cox, D. and Miller, H. D. (1965) The theory of Stochastic processes, *Chapman and Hall,* London.

[54] Cox and Isham 1986, Virtual waiting time and related processes,*Aadv. appl. Prob.* Vol18. pg.558–573.

[55] Cramer, H. (1964) Mathematical methods in statistics. *princeton University Press,*.

[56] Crosby, S., Leslie, I., Lewis, J., Toomey, F. and Davey, M. (1995) By–passing modelling: an investigations of entropy as traffic descriptor in the Fairisle ATM network. *Proceedings 12th UK Teletraffic Symposium,* Windsor, UK., IEE, Springer.

[57] Crosby, S. *et al* (1998) Measurement–based resource allocation for multimedia applications. In *Proc. Multimedia Computing and Networking,* San Jose, California. Editors Jeffar, K. *et al.* .

[58] Dai, J. G. and Harrison, J. M. (1991) Steady–state analysis of RBM in rectangle: numerical methods and queueing application. *Ann. Appl. Prob.* 1, 16–35.

[59] Dassios, A. (1987) Applications of PDP theory to queues, PhD thesis (Imperial College, University of London).

[60] Dassios, A. and Embrechts, P. (1989) Martingales and insurance, Commun. Stat. –Stochastic models, 5(2), 181–217.

[61] Davis, M.H.A. and Vinter R. B. (1984) Stochastic modelling and Control, *Chapman and Hall.*

[62] Davis, M. H. A. and Howl, J. M. (1997) A Markovian analysis of finite–buffer $M/D/1$ queue. *Proc. Roy. London.*

[63] Davis M. H. A., Dempster, M. A. H., Sethi S. P. and Vermes D., (1987) Optimal Capacity Expansion under Uncertainty, *Advanced Applied Probability,* Vol. 19, 156–176.

[64] Davis M. H. A. (1993) Markov Models and Optimization,*Chapman and Hall* .

[65] Davis M. H. A. (1984) Piecewise–deterministic Markov processes: A general class of non–diffusion models, *J. Royal Statistical Soc., Ser. B, vol.46, pp.353–388,* .

[66] Davis M. H. A. (1984) Piecewise–deterministic Markov processes: A general class of non–diffusion models, *J. Royal Statistical Soc., Ser. B, vol.46, pp.353–388,* .

[67] Dempster, M.A.H. and Ye, J. J. (1992) Necessary and Sufficient optimality conditions for control of Piece–wise Deterministic Markov processes. *Stochastics and Stochastics reports,* Vol. 40, 125–45.

[68] Derman, C (1970) Finite State Markovian Decision Processes, *Academic Press, New York.*

[69] Doop, , J. L. (1992) Renewal Theory from the point of view of the theory of probability. *Trans. Amer. Math, Soc.,* (63, 422–438, (1948).

[70] Doop, J. L. (1953) Stochastic processes. *Wiley,* .

[71] Doshi, B. T. *et al* (1978) A production– inventory control model with a mixture of back–orders and lost sales. *Management science,* Vol. 24, No. 10.

[72] Drainkov D., Hellendoom, H., and M. Reinfrank, M. (1993), An introduction to Fuzzy Control, Berlin: Springer–Verlag.

[73] Draper, N. R. and Simth H. (1998) Applied Regression analysis, *Wiley.*

[74] Dubnicki, C., Bilas, A., Li, K., "Design and Implementation of Virtual Memory Mapped Communication on Myrinet", *11th International Parallel Processing Symposium, Geneva, Switzerland,* April 1997.

[75] Dubnicki, C., Bilas, A., Chen, Y., Damianakis, S., Li, K., "VMMC-II: Efficient support for reliable, connection oriented communication", *Princeton University, CS Department,* February 1998.

[76] Duffield N. G., Lewis, J. T., O' Connell, N., Russell, R., and Toomey, F. (1995a) Entropy of ATM traffic streams. *IEEE JSAC,* 13(6): 982–990.

[77] Duffield N. G., Lewis, J. T., O' Connell, N., Russell, R., and Toomey, F. (1995b) Predicting quality of service for traffic with long–range fluctuations. In *IEEE ICC '95 Proceedings,* pages 473–477.

[78] Duffiled, N. G. abd E. Buffet, E. (1994) Exponential upper bounds via martingales for multi–plexes with Markovian arrivals. *Journal of applied Probability* , 31, 1409– 1061.

[79] Dynkin, E. B. (1965) Markov process, I and II, *Springer–Verlag, Berlin and New York,.*

[80] Eicken, T., Culler, D., Goldstein, S., Schauser, K., "Active Messages: A mechanism for integrated communication and computing", *19th Annual International Symposium on Computer Architecture,* Australia, May 1992.

[81] Engle, R. F. (1982) Autoregressive condional Heteroscedasticity with estimates of variance of United Kingdom inflation. *Econometrica,* 50:987–1007.

[82] Ethier, S. N. and Kurtz, T. G. (1986) Markov processes: Characterisation and convergence, *Wiley: New York,.*

[83] Fairid, M. (1997) Optimal Control of PDPs and Viscosity solution, PhD thesis, Imperial College, Dept. Electrical Eng..

[84] Feller, W (1951) An introduction to the Prob. Theory and its Applications *(John Wiley)*

[85] Feller, W. (1971) An introduction to probability Theory and Its Application Vol. II, 2nd. ed; *Wiley, New York,* .

[86] Fiedler, M, (1975) A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory, *Czech. Math. J.* 25:619–637.

[87] Fieller, E. C. (1932) The Distribution of the Index in a Normal Population. *Biometrika, 24, 428-440.*

[88] Field, A.J., Harrison, P.G., "An Analytical Model of the Standard Coherent Interface 'SCI'",*Proceedings of the 1995 IEEE/ACM International Conference on Parallel Processing, Oconomowoc, Wisconsin, August 1995,* pp. (I)173-177.

[89] Filex, D. P. Yager R. R. (1991) A generalised defuzzified method under BADD distributions. *Int. J. Intelligent Syst.*

[90] Fishman, G. S. 1978 Principles of Discrete Event Simu- lation. *John Wiley and Sons, New York.*

[91] Fleming, W. H. and Rishel, R. (1975) Deterministic and Stochastic optimal control, *Springer–Verlag, New York.*

[92] Fleming, W. H. and Soner, M. H. (1993) Controlled Markov processes and Viscosity solution. *Applications of Mathematics. Springer–Verlag, New York.*

[93] Fletcher, R. (1985) Semidefinite matrix constraints in optimisation, *SIAM J. Control and optimisation, Vol. 23, No4.*

[94] Fletcher, R. (1970) A New Approach to Variable Metric Algorithms, *Computer Journal,* Vol. 13, pp. 317–322.

[95] Frederiksen N. (1990) Diagnostic Monitoring of Skill and Knowledge Acquisition.

[96] Fuller, W. A. (1976) Introduction to Statistical Time Series. *John Wiley and Sons, New York.*

[97] Funahashi, K. (1989) On the approximate realization of continuous mappings by neural networks, *Neural Networks 2* .

[98] Gahinet, P., Nemirovski A., Laub A. J., Chilali M. (1995), LMI Control Toolbox, *The Math Works Inc.*

[99] Gaines, B. (1976) Foundations of fuzzy reasoning,*Int. J. Man Mach. Stud. 6* 623–688.

[100] Gavish, B. and Schweitzer, P. J. (1977) Markovian queue with bounded waiting time, *Management science,* 23, 1349– 1357.

[101] Gevers, M. and Kailath, T (1973) The discrete-time stochastic realization problem: minimum variance property of the innovations representation. *In Proc. IEEE Conf. on Dec. Contr.,* pg. 168–181, San Diego, CA.

[102] Gibbens, R. J., Kelly, and Key P. B. (1995) A decision theoretical approach to call admission control in ATM networks. *IEEE Journal of selected Areas in Communications,* 13(6).

[103] Glynn, P., and P. Heidelberger (1992) Experiments with Initial Transient Deletion for Parallel, Replicated Steady State Simulations. *Management Science, 38, 400-419.* Kelton, W. D., and A. M. Law. 1985. The Transient Behavior of the M/M/s Queue, with Implications for Steady-State Simulation. *Operations Research, 33, 378-396.*

[104] Gnedenkho, B. V. and Kovelenkho, I. I. (1966) Introduction to the theory of mass service **Russian.** *Nauka . English translation: 1968.*

[105] Golfarb, D. (1970) A Family of Variable Metric Updates Derived by Variational means *Mathematics of Computing,* Vol. 24, pp. 23–26.

[106] Golub, Gene H. and Loan, Van Charles, (1996) Matrix Computations, 3rd edition.

[107] Goodall, C. (1991) Procrutes Methods in Statistical Analysis of Shape*J. Royal Statistical Soc., Ser. B,* vol.53, pp. 285–339.

[108] S Goonatilake and P Treleaven, (Eds), "Intelligent Systems for Finance and Business", John Wiley and Sons, 1995.

[109] S Goonatilake and S Khebbal, (Eds), "Intelligent Hybrid Systems", John Wiley and Sons, 1995.

[110] Grabill, F. A. (1976) Theory and application of linear model. *Duxbury Press, Mass.*

[111] Grace, A. (1993) Optimisation Toolbox, Matlabs programing tools, *Mathworks, .*

[112] Granger, T. (1998) Measure Based Resource Allocation for Multimedia Applications. In *Proc. Workshop on Multi–Service Networks–98, Cosener's House, Abingdon, UK* 6–7 August.

[113] Green, M. and Limebeer (1995) Linear Robust Control. *Prentice Hall.*

[114] Grigor'yan, A. (1997) Analytic and geometric background of recurrence and non–explosion of the Brownian motion on Reimannian manifolds a ,*preprint, Imperial College, Mathematics Department, In Grigor'yan 's Home page, it has now appears in Bulletin of Amer. Math. Soc.* Vol. 36, pg. 135–249.

[115] Guillemin, F. and Mazumdar, R (1995) Excursions of the workload process in $G/GI/1$ queues, *Stochastic Proc. and their Appl.* Vol. 58.

[116] Guillemin, F., Mazumdar, R. and Simonian, A (1996) On heavy traffic approximations for transient characteristics of $M/M/\infty$ queues, *J. Appl. Prob.,* Vol. 33, No. 2, pg. 490–506.

[117] Guillemin, F. and Mazumdar, R. (1997) On pathwise analysis and existence of empirical distributions for G/G/1 queues, *Stoch. Proc. and their Appl.* , Vol. 67, No. 1, pg. 55-67.

[118] Habib, I. W., Ed; (1995) Neurocomputing in high– speed networks, Special issue, *IEEE Communications Magazine,* Vol.33, No.10.

[119] Hall, P. (1988) An introduction to the theory of coverage processes, *Wiley,*.

[120] M. Brown and C. Harris. (1994) Neurofuzzy Adaptive Modelling and Control, (Prentice Hall, Hemel Ham).

[121] Hastie, T. and Kishon E. (1991) A follow up discussion in (Goodall 1991).

[122] Heger D., Shah, G., "GPFS 1.4 for AIX - Architecture and Performance", *IBM White Paper,* Poughkeepsie, NY, 2001

[123] Return policies for inventory system with positive and negative demands , *Naval Research. Log. Quart. Vol. 25 581–596.*

[124] Heyman, D. P. and Sobel, M. J. (1982) Stochastic models in operational research, Vol. I *McGraw Hill, NY,* .

[125] Hille, E. and Phillips, R.S. (1957) Functional Analysis and Semigroups, *Amer. Math. Soc. Colloq.* Pub. Vol. 31, Providence R. I.

[126] Hiramatsu, A. (1990) ATM Communications network control by neural networks, *IEEE Trns. Neural Networks,* Vol. 1, pp.122–130.

[127] Hiramatsu, Atsuri (1991) Integration of ATM Call Admission Control and Link Capacity Control by Distributed Neural Networks, *IEEE Journal of selected Areas in Communications,* Vol. 9, No. 7.

[128] Hiramatsu, Atsushi, (995) Training Tecniques for Neural Networks Applications In ATM, *IEEE Communications magazine.*

[129] Hsu Ivy and Walrand Jean (1994) Dynamic Bandwidth Allocation for ATM Switches, *Journal of Applied Probability.*

[130] Hui, J. (1988) Resource Allocation for Broadband Networks*IEEE Journal of selected Areas in Communications,* Vol. 6 No. 9.

[131] IEEE, "Scalable Coherence Interface Draft 2.0", March 1992.

[132] Iserles A. (1993) On the generalized pantograph functional–differential equation. *European Journal of Appl. Maths,* Vol 4. 1–38.

[133] Iserles A. (1996) A first course in Numerical Analysis of differential equations. *Cambridge University Press, Cambridge.*

[134] ITU Publications, (1997) ITU World Telecommunication Indicators Database, Federal Communications Commission, *http: //www.itu.org/publcations/*

[135] Jacobson, D. H. and Mayne, D. Q. (1970) Differential Dynamic programming, *Elsevier.*

[136] Jang, J. S. R. (1992) Adaptive-Network-based Fuzzy Inference Systems, *IEEE Trans. Systems, Man and Cybernetics,* Vol..23, No.3, pp.665.

[137] Jang, J. S. R. (1993) Self–learning of fuzzy controller based on temporal back propagation,*IEEE Trans. Neural Networks,* Vol. 3, No.5.

[138] Kac, M. (1947) Random walk and the theory of Brownian Motion, *American Mathematical Monthly,* Vol. 54 pp. 369- 391.

[139] Kalman, R. E. (1982) System identification from noisy data, In *Dynamical systems II, (A. R. Bednarek and L. Cesari, Eds.), Academic press, New York,* pp. 135–164.

[140] A. Kandel , A. (1994) Fuzzy control systems, *CRC Press.*

[141] Karmarkar, N. K.and Ramakrishnan, K. G. (1991) Computing results of interior point algorithm for large scale linear programming . *Mathematical programming,* Vol. 52 No. 3 pg. 555– 586.

[142] Kelly, F. P. (1984) In the discussion of Davis's paper (Davis 1984), Frank Kelly complained that the main reason the work of (Cox 1955) and (Gnedenkho 1966) has not been taken seriously before Davis' work on PDPs is the absence of a general methods for dealing with the integro–differential equations which arise.

[143] Kelly, F. P. (1991) Effective bandwidths at multi– class queues. *Queueing systems,* 9, 968–981.

[144] Kelly, F. (1996) Notes on effective bandwidth, paper is published in *Stochastic networks, Editors: Kelly, F, Zachary, S, Vol. 4 Royal Statistical Society Lecture Notes, 141–168.*

[145] Kemeny, J. G., Snell, J. L. and Knapp, A. W. (1966) Denumerable Markov Chains. *Van Nostrand, New York, .*

[146] Kendall, D. G. (1953) Stochastic processes occuring in the theory of queues and their analysis by the means of the imbedded Markov chain. *Ann. Math. Stat. Vol. 24, 338–354.*

[147] Kingman, J. F. C. (1964) A Martingale inequality in the theory of queues.*Proc. Cam. Phil. Soc.,* 59, 359–361.

[148] Kingman, J. F. C. (1972) Regenerative phenomena.

[149] Kingman, J. F. C. (1976) Anticipation processes. In *Pespectives in Probability and Statistics,* (Gani, J. Editor, New York Applied Prob. Trust.)

[150] Koenigs, A. "Industrial Strength Parallel Computing",*Morgan Kaufmann,* San Francisco, 2000

[151] Kohonen, T. (1984/1989), Self-Organization and Associative Memory, edn, Springer-Verlag, Heidelberg.

[152] Kohlas, J. (1982) Stochastic Methods of Operations Research, *Cambridge University Press, London,.*

[153] H. M. Kim H. M, B. Kosko, (1992) Fuzzy prediction and filtering in impulsive noise, *Fuzzy Sets and Systems 77 .*

[154] Kosko, B. (1992) Fussy systems as universal approximators. *Proc. IEEE Int. Conf. Fuzzy syst. San Diego,* 1153- -1162.

[155] Kim, H. M., and Kosko, B. (1996), Fuzzy prediction and filtering in impulsive noise, *Fuzzy Sets and Systems* , 77.

[156] Hartman E. J., Keeler J. D., Kowalski J. M., (1990), Layered Neural Networks with Gaussian Hidden Units as Universal Approximations, *Neural Computing,* Vol. 2, 210-215.

[157] Kulkarni V. G. (1995) Modeling and Analysis of stochastic systems. *Chapman and Hall..*

[158] Kumar, P.R. and Varaiya. (1986), Stochastic systems, *Prentice Hall.*

[159] Lakshmikantham V. and Leela S. W.(1969) Differential and integral inequalities, Vol.1 and Vol.2 *New York: Academic Press.*

[160] Law, A. M. (1977) Confidence Intervals in Discrete Event Simulation: A Comparison of Replications and Batch Means. *Naval Logistics Review Quarterly, 24, 667-678.*

[161] Law, A. M., W. D. Kelton, (1991) Simulation Modeling and Analysis, *McGraw-Hill, New York.*

[162] Lawley, D. N. (1967) Some new results in maximum likelihood factor analysis, *Proc. Roy. Soc. Edinb;* A Vol67, 256–264).

[163] Lawley, D. N. (1956) Tests of significance for latent roots of covariance and correlation matrices, *Biometrika,* Vol. 43, pp.128–136.bibitemMaxwell71 Maxwell and Lawley Lawley, D. N. (1971) Factor analysis and statistical method. *Butterworth and Co.*

[164] Lax P. D. (1968) *Pure and applied maths,* Vol. 21, 467.

[165] Levenberg, K. (1944) A method for the solution of certain problems in least squares, *Quart. Apl. Math.* Vol.2, pp. 164–168.

[166] Lightbody G., and Irwin G. W. (1995), Direct neural model reference adaptive control, *IEE Proc.* 142(1).

[167] Linkens D. A., and Nie J. (1992) A unified real-time approximate reasoning approach for use in intelligent control Part 1 and Part 2, *International Journal of Control,* Vol.56, No.2.

[168] Ljung, L. (1977) Analysis of recursive stochastic algorithms, *IEEE Transactions on Automatic Control,* AC–22(4) 207–220.

[169] Sjoberg J. And Ljung L. (1995) Over training, regularisation, and searching for minimum in neural networks, Preprint *IFAC Symposium on Adaptive Systems in Control and Signal Processing. Grenoble, France,* pp.669-674.

[170] Lord, F. M. and Novick, M. R. (1968) Statistical theory of mental test scores (with contributions by A. Birnbaum) *Addison–Wesley, Reading, M.A. .*

[171] Lorentz, G. G. (1986) Approximations of Functions. Chelsea Publ. Co., New York

[172] Luenberger, D. G. (1969) Optimisation with vector space methods, Wiley,.

[173] Lundberg, J. (1930) Uber die Wahrscheinlichkietsfunktion einer Risikenmasse. *Skandinavisk Aktuarietidskrift* 13, 1–83.

[174] Maana, A. (1997) M. Phil Thesis, Sept. 1999, Supervisor Susan Pitts. *Statistical laboratory in Cambridge University.*

[175] Maglaris, B., Anastassiou, D. Karlsson, G. P., Robins, J.(1988) Performance models for statistical multiplex in Packet Video Communications,*IEEE Trans. on Communications,* Vol. 36,pp. 834–844.

[176] Mallison, H. and Bentley, P. J. (1999) Evolving fuzzy rules for pattern classification . In *Proc. of Int. Conf. on Computational Intelligence for modelling control and Automation. CIMCA'99.* Also, check these references in their home pages in *Computer science in University College London.*

[177] Assilain, S. and Mamdani E. H. (1976) Advances in linguistic synthesis with fuzzy logic controller, *Int. J. Man, Machine- Studies,* , pp. 669–678.

[178] Marquardt, D. (1963) An algorithm for the least squares Estimation of Nonlinear parameters, *SIAM J. Appl. Math.* Vol. 11, pp. 431–441,.

[179] Matlab (1995) Matlabs programing tools, *Mathworks,* .

[180] McBride, A. C. (1987) Semigroups of Linear Operators: an introduction, *Longman Scientific and Technical,* .

[181] McDysan D. E., ATM theory and Application, (1994) *Mcgraw–Hill Series Computer Communications.*

[182] MacKay, D. J. C. (1992) A Practical Bayesian Framework for Backpropagation Networks. *Neural Computation* , Vol. 4 , pp.448–472.

[183] Wang L. X. and Mendel, J. M. (1992) Fuzzy basis functions, universal approximation, and orthogonal least square learning,*IEEE Trans. Neural Networks 3* 807-814.

[184] Montroll, E. W. (1965) Random walks on Lattices. *II Journal of Mathematical Physics,* Vol. 6 No. 2,

[185] Moody, J., And Darken, C. (1989) Fast learning in networks of locally–tuned processing units, *Neural Comp. 1* , 281-294.

[186] Morrison, D. F. (1967) Multivariate statistical methods, *McGraw–Hill Series, New York.*

[187] Morse, P. M. (1958) Queues, Inventories and Maintenance, *John Wiley and Sons, INC,* .

[188] Mukherjee, B. N. (1966) Derivations of the likelihood ration tests for Guttman quasi–simplex covariance structures, *Psychometrica,* Vol. 31, pp. 97–123.

[189] Murray, J. R., and W. D. Kelton (1988). The Transient Behavior of the M/E2/2 Queue and Steady State Simulation. *Computers and Operations Research, 15, 357–367.*

[190] Narendra K. S. And Parthasarathy (1990) Identification and Control of dynamical systems using neural networks, *IEEE Trans. Neural networks,* Vol1. No.1. 4-27.

[191] Ndouse T. D. (1994) Fuzzy Neural control of voice cells in ATM Networks, *IEEE. J. Sel. Areas in Communications.* Vol. 12. No. 9.

[192] Nesterov Yu and Nemirovski A. (1994) Interior–point polynomial methods in convex programming, Volume 13 of Studies in Applied Mathematices,

[193] Nie, J., And Linkens, D. (1993) Learning Control Using Fuzzified Self-Organising Radial Basis Function Network, *IEEE Trans. of Fuzzy systems,* Vol.1, No.4.

[194] Odoni, A. R. E. Roth, E. (1983) An Empirical Investigation of the Transient Behavior of Stationary Queueing Systems. *Operations Research, 31,432-455.*

[195] Onvural, R. M. (1995) ATM networks: Performance analysis.

[196] Pan, H. and Wang, I. Y.(1991) The bandwidth allocation in ATM through Genetic Algorithms, *GLOBECOM 91, session 4.4, pp.125–129.*

[197] Pawlikowski, K. 1990. Steady-State Simulation of Queueing Processes: A Survey of Problems and Solu-tions. *ACM Computing Surveys, 22, 123-171.*

[198] Pazy, A. (1983) Semigroups of Linear Operators and Applications to Partial Differential equations. *Springer-Verlag.*

[199] Fuzzy control and fuzzy systems. *Wiley and Son, New York.*

[200] Pegden, C. D., Sadowski, Sadowki and Shannon, R. E. (1990), An introduction yo simulation using SIMAN, *Systems Modeling Corporation, Sewickley, PA. USA.*

[201] Pitsillides A., Sekercioglu Y. A.,and Rammurthy G.(1997) Effective Control of Traffic Flow in ATM Networks Using Fuzzy Explicit Rate Marking (FERM) *IEEE. J. Sel. Areas in Communications.* Vol. 15. No. 2.

[202] Pitts S. M. (1993) Non–Parametric estimation in renewal theory, I. the empirical renewal function. *Ann. Stat.;* 21, 1431–1451.

[203] Bingham N. H. and Pitts S. M. (1998) Non–Parametric inference for the $M/G/\infty$ queue. To appear in *Institute of Statistical mathematics,*

[204] Poggio, T., and Girosi, F. (1990a) Regularisation algorithms for learning that are equivalent to multi layer networks, *Science,* 247, 978-98

[205] Poggio, T., and Girosi, F. (1990b) Networks for approximation and learning, *Proc. IEEE ,* Vol. 78, pp. 1481-1497

[206] Pothen, H. Simon, and K.-P. Liou (1990), "Partitioning sparse matrices with eigenvectors of graphs", *SIAM J. Matrix Anal. Appl.,* 11:430–452.

[207] Powell, M.I.D. (1985) Radial basis function for multivariable interpolation: a review,*IMA Conference on Algorithms for the approximation of functions and data, RMCS,* Shrivenham.

[208] Prylli, L., Tourancheau, B., "Protocol Design for High Performance Networking", *LIP-ENS, Lyon,* July 1997.

[209] Rockafellar, R. T. (1970) Convex analysis *Princeton, N.J: Princeton University Press.*

[210] Ross S. M. (1970) Applied probability Models with Optimisation Applications. *Holden–Day, San Francisco,* .

[211] Ross S. M. (1983) Stochastic Processes, *Wiley Series in Probability and Mathematical statistics* .

[212] Ross S. M. (1983) Stochastic Processes, *Wiley Series in Probability and Mathematical statistics* .

[213] Roth, E. (1994) The Relaxation Time Heuristic for the Initial Transient Problem in M/M/k Queuein,g Systems.*European Journal of Operational Research,* 72, 376-386.

[214] Saad, Y. (1992) Numerical methods for large eigen value problems. *Halstead Press, NY.*

[215] Sargent, R. G., Kang, K. and D. Goldsman, G (1992) An Investigation of Finite-Sample Behavior of Confidence Interval Estimators. *Operations Researchl, 40, 898-913.*

[216] Schruben, L. (1982) Detecting Initialization Simulation Output. *Operations Research, 30,569-590.*

[217] Seal, H. L. (1969) Stochastic theory of risk business. New York: Wiley.

[218] Seber, G.A.F. and Wild, C. J. (1989) Nonlinear regression, *Wiley series in Probability and mathematical statistics.*

[219] Segall A. and Kailath, T (1973) A further note on innovations, martingales and nonlinear estimation. In *Proc. IEEE Conf. on Dec. and Contr.,* pages 616–620, San Diego, CA.

[220] Shah, G., Nieplocha, J., Mirza, J., Kim, C., Harrison, R., "Performance and Experience with LAPI - a New High-Performance Communication Library for the IBM RS/6000 SP", *International Parallel Processing Symposium,* Orlando, FL, 1998.

[221] Shanno, D. f. (1970) Conditioning of Quasi–Newton Methods for Function minimisation, *Mathematics of Computing,* Vol. 24, pp. 647–656,

[222] Shavitt, I. S. (1963) The guassian function in calculations of statistical mechanics and quantum mechanics, in *Methods of computational Physics: Quantum mechanics,* Vol.2, pt1, B. Alder, S. Fernbach, and M. Rottenberg, eds. Academic Press, New York.

[223] Sinclair, Mark (1993) The application of Genetic Algorithm to Trunk Network Routing Table optimisation, *IEE Colloquim (Digest) UK Teletraffic Symposium on performnce Engineerin in Telecommunications,* .

[224] Sleeman , D. (1984) An attempt to understand student's understanding of basic algebra " . 1984 . Cognitive Science

[225] Sleeman , D. (1989) Studies and diagnosis with high school algebra students. *Cognitive Science.*

[226] Slotine, J. J., and Li, W. (1991) Applied non–linear control, *Prentice Hall International,* Englewood Cliffs , NJ

[227] Sanner, R. M., And Slotine, J. J. E. (1992) Guassian Networks for Direct Adaptive Control, *IEEE Trans. of Neural Networks,* Vol. 3, No. 6.

[228] Snell, M. and Schruben, L. (1984) Weighting Simulation Data to Reduce Initialization Effects. *IIE Transactions, 17,354-363.*

[229] Spat, H. (1980) Clustering analysis algorithms. *Ellis Horwood Ltd.*

[230] Stoer, J. and Witzgall, C.(1970) Convexity and Optimisation in Finite Dimensional I, *Springer-Verlag* .

[231] Stroock, D. and Varadhan, S. R. S. (1976) Multidimensional Diffusion Processes. *Berlin, Heidelberg and New York: Springer–Verlag..*

[232] Suen H . K (1990) Principles of Test Theories.

[233] Takagi and Sugeno, M. (1983) Derivation of fuzzy rules from human operator's actions, *Proc. IFAC Symp. Fuzzy Information, Knowledge Representation and Decision Analysis.*

[234] Takacs, L (1955) Investigation of waiting time problems by reduction into Markov processes. *Acta Math. Acad. Sci. Hun.* Vol. 6 pp101-129.

[235] Tanaka, H. (1963) A note on continuous functionals of 1–dimensional Brownian motion Path. *Z. Wahrsheinlichkeitstheorie verw. Gebiete 1,251–257* .

[236] Tankeh, N. A. (1989) Efficient neural networks training, MSc Thesis, Control Group, Department Electrical Engineering, Imperial College, London.

[237] Tankeh, N. A. (1993) Application of Item Response Theory (IRT) to educational testing of students knowledge of algebra. PGCE Thesis for Postgraduate Certificate fo Mathematical Education (PGCE). Department of Mathematics, Institute of Education, University of London.

[238] Tankeh, N. A. and Mamdani, E. H. and Azvine, B. (1997a) Soft Computing with Markov Chains and Convex optimisation. In Proc. of *Second Intern. Computer Conventions (ICSC) Symposia on Intelligent Industrial Automation and Soft Computing.* Held in France, Sept.

[239] Tankeh, N. A. and Mamdani, E. H., (1997b) Relative merits of conjugate gradient algorithm in Soft Computing Learning. In *Proceedings of Firth International Conference on fuzzy theory and Technology.* Durham, U.S.A.

[240] Tankeh, N. A. and Mamdani, E. H., (1997c) Convex fuzzy restriction, *Proc. of International conf. on computational intelligence, Fuzzy Days, Dortmund, Germany,* .

[241] Tankeh, N. A. and Mamdani, E. H., (1997d) Convex fuzzy Controller, In *Proc. of 6th IEEE conf. in fuzzy systems.*

[242] Tankeh, A. T. and Mamdani, E. H. (1998) Martingale model of ATM, *Proc. IFIP conference on ATM networks, Ilkley, Bradford, West Yorkshire,* 20th–22nd July.

[243] Tankeh, A. T. (1998) ATM as a renewal process. In *Proc. Workshop on Multi–Service Networks–98, Cosener's House, Abingdon, UK* 6–7 August.

[244] Tapiero, C. (1990) Applied Stochastic models and control in management, *North–Holland.*

[245] Tatsuoka , K. K .(1983) Rule space approach for dealing with misconceptions based on Item Response Theory . *Journal of Educational Measurement,* Vol 20 ,34–38.

[246] Tatsuoka , K. K. (1985) A probabilistic model for diagnosing misconceptions by the pattern classification approach. *Journal of Educational Statistics,* Vol . 50 ,55–73

[247] Tatsuoka K . K. (1985) A probabilistic model for diagnosing misconceptions by the pattern classification approach. *Journal of Educational Statistics,* Vol . 10 , No. 1.

[248] Tezuka, H., O'Carrol, F., Hori, A., Ishikawa, Y., "A Virtual Memory Management Technique for Zero-Copy Communication, 12th International Parallel Processing Symposium, Orlando, FL, April 1998.

[249] Thorin, O. (1982) Review, *Scandinavian Actuarial Journal.* .

[250] Tijms H. C. (1975) On switch–over Policy for controlling the workload in queueing Systems with Two constant Service time rates and Fixed Switch–Over Costs. *Z. Operations Res.* Vol.21, 19–32.

[251] Titterington D. M. (1975) Optimal design: some geometric aspects of D–optimality.

[252] Titterington, Titterington D. M. (1977) Estimation of correlation coefficients by Ellipsoidal Trimming.

[253] Tjims, H. C. (1986) Stochastic Modelling and Analysis: A computational Approach, *Wiley,*.

[254] Meyn, S. P., and Tweedie R. L. (1992) Stability of Markov processes. I. Criteria for discrete time chains. *Adv. Appl. Prob.,* Vol. 24, 542–574.

[255] Faraday project in Computer science Department of UCL *http: kdd.ics.uci.edu.*

[256] Ventsel A. D. (1961) Additive functionals of several dimensional Wiener process,*Doklady Akad. Nauk SSSR n. Ser. 139, 13–16* .

[257] Vylder, F. (1977) Martingales and Ruin in dynamical Risk Process, *Scand. Actuarial J.* 217–225.

[258] Walrand, Jean and Varaiya, Pravin (1996) High Performance Communication Networks, *The Morgan Kaufman Series in Networking* .

[259] Wald, A. (1947) Sequential analysis, *Dover Publications, New York,*.

[260] Walrand, J. (1988) An introduction to queueing networks, *Prentice–Hall, Englewood Cliffs, NJ.*

[261] Walter, W.(1964) Differential and integral inequalities, *Ergebnisse Der Mathematik Und Ihrer Grenzgebiet.* Band 55.

[262] Wang A. T. (1977) Generalised Ito's formula and additive functionals of Brownian motion.*Z. Wahrsheinlichkeitstheorie verw. Gebiete 41,153–159* .

[263] Wang, L. X. and Mendel, J. M. (1992) Fuzzy basis functions, universal approximation, and orthogonal least-squares learning, *IEEE. Trans. Neural Networks,* 3(5), pp.607-814.

[264] Warschko, T., Blum. J., Tichy, W., "Design and Evaluation of ParaStation2", *International Workshop on Distributed High Performance Computing,* Essen, 1998a.

[265] Warschko, T., Blum. J., Tichy, W., "On the Design and Semantics of User-Space Communication Subsystems, *University of Karlsruhe,* 1998b.

[266] Wax, H. and Kailath, T. (1985) Detecting signal by information theorical Criteria, *IEEE Trans. Acoustic. Speech Signal Proc.* Vol. 33, pp. 387–392.

[267] Nguyen D. H., and Widrow B. (1990) Neural networks for self-learning control systems, *IEEE Contr. Syst. Mag.*

[268] Wolff, R. W. (1989) Stochastic Modeling and the Theory of Queues, *Prentice–Hall, Englewood Cliffs, NJ,* .

[269] Woodgate, K. G. (1995) An upper bound on the number of linear relationships identified from noisy data using the Frisch scheme, *Systems and control letters,* Vol. 24, pp.153–158. *SIAM, Philadelphia, PA, USA.*

[270] Yager,R. R. and Zadeh, L. A. (1994) Fuzzy sets, Neural networks and Soft Computing. *Van Nostrand Reinhold* .

[271] Yager,R. R., Ovchinniko, S. Tong, R.M and Nguyen, H. T. (1987) Fuzzy sets and its applications, *Selected papers by Zadeh , John Wiley and Sons* .

[272] Yazdi S. *et al* (1992) Congestion control methods for B–ISDN, *IEEE Commun. Mag.* (Vol. 30, N0. 7. pp.42–47.

[273] Youssef, S. A., Ibrahim, W. and Habib, W. (1997) A neurocomputing Controller for bandwidth Allocation in ATM networks, *IEEE. J. Sel. Areas in Communications.* Vol.15. No.2.

[274] Zachmanoglu, E. C. and Thoe, D. W. (1986) Introduction to partial differential equations with applications, *New York: Dover,* .

[275] Zacks, S. (1971) The theory of statistical inference. *John Wiley, New York.* .

[276] Zadeh, L. A. (1973), Outline of a New approach to the Analysis of Complex Systems and Decision Processes, *IEEE Trans. Systems, Man, and Cybernetics,* SMC-3, 28-44.

[277] Zadeh, L. A. (1975), The Concept of Linguistic Variable and Application to Approximate Reasoning, Part 1,2 and 3, *Information Sciences,* 8.

[278] Zadeh, L. A. (1976), A Fuzzy-Algorithmic Approach to the Definition of Complex or Imprecise Concepts, *Int. J. Man- Machine Studies,* 8.

[279] Zadeh, L. A. (1978), A Fuzzy–set–Theoretic interpretation of linguistic hedges, *J. Cybern,* 4-34

[280] Bellen A., Guglielmi N. and Zennaro A. (1999) On the contractactivity and asymptotic stability of the systems of delay differential equations of neutral type, *BIT,* Vol. 39, No. 1 pp. 25–33.

BIBL. LONDIN. UNIV.