Spatial Reaction Systems on Parallel Supercomputers

Mark Smith

Doctor of Philosophy

1994

Abstract

A wide variety of physical, chemical and biological systems can be represented as a collection of discrete spatial locations within which some interaction proceeds, and between which reactants diffuse or migrate. Many such real-world spatial reaction systems are known to be both non-linear and stochastic in nature, and thus studies of these systems have generally relied upon analytic approximation and computer simulation. However, this later approach can become impractical for large, complex systems which require massive computational resources.

In this work we analyse a general spatial reaction system in both the deterministic and stochastic scenarios. A study of the deterministic parameter space reveals a new categorisation for system development in terms of its *criticality*. This result is then coupled with a complete analysis of the linearised stochastic system, in order to provide an understanding of the spatial-temporal covariance structures within reactant distributions. In addition to an analysis, and empirical confirmation, of the various criticality behaviours in both deterministic and stochastic cases, we use our theoretical results to enable efficient implementation of spatial reaction system simulations on parallel supercomputers. Such novel computing resources are necessary to enable the study of realistic-scale, long-term stochastic activity, however they are notoriously difficult to exploit. We have therefore developed advanced programming and implementation techniques, concentrating mainly on dynamic load-balancing methodologies, to enable such studies. These techniques make direct use of our analytic results in order to achieve the most efficient exploitation of supercomputing resources, given the particular attributes of the system under study. These new techniques have allowed us to investigate complex individual-based systems on a previously untried scale. In addition, they are of general applicability to a wide range of real-world simulations.

This work is dedicated to my parents,

their unfailing encouragement and support got me to it,

and to my wife, Ute,

her amazing patience, help and love got me through it.

ἀγεωμέτρητος μηδεὶς εἰσίτω

Declaration

I confirm that the work detailed in this thesis constitutes all my own efforts, and that this thesis has been composed wholly by myself.

Mark Smith

Acknowledgements

This thesis could not have been completed without the help and support of a great many people. I am deeply indebted to Professor Eric Renshaw. Without his support, encouragement and belief this work may well have floundered in its infancy. I hope that all the pain and frustration that I may have caused do not deter him from taking on similar students in the future.

I must also acknowledge the outstanding support that I have received from the Edinburgh Parallel Computing Centre over the past four years. Their help goes far beyond the provision of an excellent working environment and access to their *unparalleled* resources. My friends and colleagues in the Centre have always encouraged my studies, and have also provided me with the motivation and knowledge to enable them. Out of so many that have helped me, a number stand out for mention: Nick Radcliffe for his knowledge of so many things, and his willingness to answer my pestering questions on all of them; Kevin Collins for being such an enthusiastic and understanding office-mate; Neil Heywood for allowing me to bend so many rules; Neil MacDonald for his technical knowledge and our competition; Matthew White and Gordon Cameron for their support of my graphical adventures; Billy Taylor for answering so many of my dumb systems questions; David Wallace and Greg Wilson for passing on their enthusiasm for parallel computing to me; the list goes on and on...

Contents

1

Introduction

I am never satisfied until I have said as much as possible in a few words, and writing briefly takes far more time than writing at length.

– Carl Friedrich Gauss

1.1 Introduction to Spatial Reaction Systems

There are a vast number of real-world systems that can be thought of in terms of an array of locations within each of which some interaction proceeds in time. In addition, the reactants involved might be mobile, and therefore migrate or diffuse between adjacent locations. Such a system can be used to model the flow of a fluid over an aerofoil; the movement of petroleum through porous rock; the distribution of chemicals within a group of cells; the movement of infected individuals and hence of a viral infection through a distributed population; or even the spatial development of competing species in an ecological model. We have attempted to bring the general cases of all such systems together under one name, and refer to them as *spatial reaction systems*. The term "spatial" is intended to convey the fact that our systems are composed of discrete locations between which individual reactants can migrate, and within which they can evolve.

The serious mathematical study of spatial reaction systems has its major roots in the pioneering work of Alan Turing during the 1950s. His particular interest lay in understanding the biochemical processes behind what he termed *morphogenesis* – the process whereby a living body takes on a particular shape, structure or pattern, either from some initial homogeneous start, or possibly following some external interference. Turing postulated that such structural formation could be triggered by the presence of certain chemicals (morphogens) in particular concentrations. His model of a developing foetus therefore consisted of a set of discrete cells within which any number of morphogens could interact, and also between which the morphogens could migrate in some manner (e.g. by diffusion or osmosis). Taking the simple case of a ring of cells, Turing [1952] linearised some general interaction functions to provide an analytic solution for the deterministic cell concentrations of each morphogen through time. His results show the formation of spatial fluctuations in morphogen concentrations as sinusoidal waves around the ring of cells.

Since Turing's initial work, the spatial reaction model (sometimes referred to as reaction-diffusion, although this term is only strictly correct for continuous systems) has grown to be an accepted standard basis for analysis and simulation in many branches of biology (see Renshaw [1986]), ecology (Smith [1991]) and zoology (Murray [1989]). It has also gained popularity in such diverse areas as petroleum engineering (see Wheeler [1988])

and epidemiology (Mollison [1977]). Whether we consider the physical, chemical or biological system at the centre of our studies as being continuous, or constructed of discrete spatial locations, a computer simulation will generally contain some level of discretisation to allow computer implementation. This can sometimes lead to a confusion between the terms reaction-diffusion and reaction-migration. It is our belief that many real-world systems are specifically discrete in their structure, and we will highlight results that expose the importance of considering them as such. Additionally, we can always return our spatial reaction systems to the continuous diffusion domain under the standard computational approximation of very many spatial locations on a fine-grained lattice.

In general such systems can be used to model any spatial system where reactants have both a particular location within which they interact, and neighbouring locations to which they can migrate. In population modelling, for example, the reactants may be different species (e.g. predator and prey), and the location is a particular geographical region of an appropriate scale, within which the species live and die with rates regulated by the local population size. We can then construct one-, two- or possibly three-dimensional models of a "global" system by specifying the spatial relationship between individual locations. The dimensionality of the *world* is dependent on the particular system under study; thus a one-dimensional system could be used for modelling movement within a river or along a coastline, a two-dimensional system could be used for land-based models, and a three-dimensional system could represent air or sea based worlds. To take an example from a different field; in petroleum engineering spatial reaction systems are used to model the mixing of water and oil in porous rocks, and the resulting flow through the rock structure. In this case the reactants are water, oil and the rock itself. Thus at each point in the model the water and oil interact with each other and also with the rock. This then affects the movement of the mobile substances through the rock, and leads to a technique for simulating oil reservoir behaviour.

These examples are obviously specific to certain fields, and such models will undoubtedly contain some difference in their system parameters and spatial structure. For our analytic studies (in particular those covered in Chapters 2 and 4) we will retain as much generality as possible in our mathematical specification. This will enable the extraction of generic descriptions of system behaviour. However, as we present results for our numerical solutions and computer simulations, we will concentrate on a partic-

ular suite of biological and ecological systems. This focus provides for ease and clarity of comparison between simulations, as well as following the subject area of Turing's original work.

1.2 Biological and Ecological Systems

Although there has been only a small amount of further study into the field initially explored by Alan Turing, that which is documented lies almost completely within the biochemical sciences. It is in this area that we can observe the most direct link between real applications and the original *discrete cell* model. However, recent research trends seem intent on pulling studies back to the continuous system. One of the first studies to review Turing's results came after a delay of over 20 years (see Bard & Lauder [1974]) and, significantly, made use of the emerging computer technology in an attempt to discredit Turing's conclusions. Bard & Lauder observed that for small numbers of cells and non-linear interactions, Turing's approximate solutions become dependent upon initial conditions. It is unfortunate that this observation then set the tone for further work, with Murray [1981] using it as evidence that discrete models are "incorrect", and hence as a reason for concentrating study on continuous diffusion systems. What is even more unfortunate is that prior to both these works, a study by Wolpert [1969] had claimed that Turing's approach was in fact too *continuous* to handle situations where spatial differentiation is required to produce different results in different regions of the model. Wolpert was interested in whether genetic information was responsible for this cellular differentiation, and developed a *gradient approach* model in which morphogen concentration varied spatially in order to control system development. This idea has been recently resurrected by Maini *et al* [1992] to investigate the large number of biological systems that are known to exhibit spatial inhomogeneity. This recent work uses a version of Wolpert's space-dependent diffusion parameter in a continuous spatial domain. In Chapter 3 we will highlight the importance of considering the discrete version of such systems, as often this conversion can provide a more straightforward solution to the inadequacies of current continuous-space diffusion models.

Some time before Bard & Lauder [1974] restarted interest in this field, a line of research into the location of hair follicles on *Drosophilia* had suggested Turing-type mechanisms

as being responsible, see Maynard-Smith & Sondhi [1961]. Standard linear interactions were used in a two-dimensional domain of cells, and this produced regular *hot-spots* in chemical (morphogen) concentration. If such high concentration locations are considered as the starting points for hair growth, patterns similar to those in nature are obtained. Further work by Nagorcka & Mooney [1985] suggests the possibility of several waves or phases of growth, and this brings the resulting theoretical patterns even closer to the real world. Our own results reviewed in Sections 3.6 and 5.4 show these same patterns, but with the addition of considering a stochastic environment. The link between spatial reaction systems and *Drosophilia* is further strengthened by Kauffman *et al* [1978] who use standard linear interactions coupled with a diffusion mechanism to explain the formation of body-parts for different functions within an embryo. This idea is taken further by Bunow *et al* [1980] by incorporating non-linear interactions (and hence using computer simulation rather than mathematical analysis) to show that a series of binary decisions based on chemical concentrations within cells can produce natural patterns. However Bunow *et al* were concerned with their results, since the use of non-linear interactions had produced "unreliable and patchy" systems. This concern has recently been countered by Hunding [1990] through incorporating Wolpert's ideas on spatially-varying parameters — something Hunding terms a "Turing pattern of the second kind".

However, the most widespread use of Turing-type systems has probably been in the study of pattern formation, particularly in animal coat markings. Again all the work in this field has been with deterministic continuous (diffusion) systems, and Murray [1989] provides an excellent review of the topic. Previous work ranges from the early analysis of the possible parameter space for Turing-pattern formation (see Murray [1982]), to the more recent and more complex simulations of gradient-diffusion models (see Nagorcka & Mooney [1992]) that show how patterns can change significantly upon a single animal. The one remaining problem in this field of research is that the actual morphogens themselves seem to avoid identification; although two very recent studies from the chemical physics discipline (Castets *et al* [1990] and Ouyang & Swinney [1991]) have shown reaction-diffusion patterns occurring under laboratory conditions.

Our particular interest in the analysis and simulation of spatial reaction systems is to assist in the study of complex ecological systems, and ultimately to understand the effects of mutation and evolution within such systems. We therefore intend to implement

complex numerical simulations of interacting species, where certain attributes of each species can be successively adapted through random mutation. We are therefore dealing at a level where it is natural to consider discrete systems. However the results we achieve can be applied directly back to those smaller-scale systems currently analysed with diffusion techniques. Such an approach can therefore satisfy the ecologist studying species migration patterns, the genetic biologists looking for waves of *gene-gap* morphogens (see Hunding [1990]), or the zoologist attempting to explain animal coat patterns.

Our study of such spatial reaction systems starts with an analysis of the deterministic linearised and non-linear systems; we then advance to the stochastic versions of these models; and thus produce measures of their expected behaviour. This is vital as we then go on to presents the results obtained through extensive simulation of these systems on the latest generation of high performance computers. In order to run *realistic* models we need to consider stochastic simulations of non-linear interactions on many sites and over long time-scales, and thus very powerful computers are an absolute requirement. This leads us to an understanding of effects introduced by non-linearity, stochasticity or discreteness in the system. This is important since a complete analysis is only available for deterministic linearised systems.

1.3 Modelling Evolutionary Processes

Charles Darwin's theory of evolution through natural selection [1859] is now well over 100 years old. It has survived many attempts, both biological (see Denton [1985]) and mathematical (Hoyle [1987]), to discredit it, yet its status in the scientific community currently seems to be at its strongest. Following the successful coupling with genetic theory by Fisher [1930], and the more recent popularisation brought about by Dawkins [1976, 1986], the theory is now as established in biology and ecology as quantum mechanics is in physics. Moreover, evolutionary analogies are also appearing in other branches of science. Neuro-scientists suggest that neuron growth in the brain occurs in a Darwinian manner, with mass-multiplication of cells followed by selection of certain connections (see Edelman [1989]). These ideas tie in well with neural network theories where systems of electronic *neurons* are thought of as "learning" solu-

tions to problems. In a recent book, Basalla [1988] suggests that evolution occurs in technological advancement in the same way as it does in natural selection. He proposes that inventors tinker with existing tools and devices, producing continuous incremental improvements, rather than the stereotypical image of the scientist or engineer making a great breakthrough. This idea can be supported by archaeologists who have searched in vain for any discontinuity in the history of the development of stone-age tools. The concept of non-biological evolution has recently been taken one step further by Root-Bernstein [1989] who presents a specific analogy between evolution and scientific discovery. He suggests that ideas themselves are passed down through generations from teachers to pupils. Thus they proliferate, possibly undergoing slight mutation, and a small number are then selected for use. These ideas lie very close to those of *memes*, as suggested by Dawkins in the final sections of *The Selfish Gene* [1976]. However, the connection between general science and evolutionary theory is probably seen most clearly in the application of genetic algorithms to problem solution (see Holland [1975]). Here multiple possible algorithms for the solution of a problem are programmed into a computer and are then allowed to mutate and inter-breed in order to provide an optimal result. A cost-function is used to describe the performance of each algorithm. The best performers have the highest chance of surviving to successive generations and thus being the "parent" of subsequent solutions. After many generations the programmer can then select the best performing algorithm from the pool of current solutions.

Evolutionary theory and ideas are clearly becoming ever more popular with the scientific community. There are, however, two remaining obstacles for those wishing to produce real evidence that evolution can account for much of what we observe in nature. First, the very essence of natural selection suggests that its effects cannot be observed in any reasonable time-scale, such that it can be analysed whilst in progress. It would therefore be ideal if computer models could be developed to reproduce the effects of evolution in a much shorter time. The recent work of Dawkins [1986] attempts to do this, and shows that very simple rules and starting configurations can produce the most intricate of final states. However, this model highlights the second obstacle, in that all the manipulation is deliberately controlled by a human operator. Each successive generation is specifically chosen with some goal in mind. There is therefore no *natural* selection through interaction with an environment, or through simple random changes. This exposes a major problem with computer models for evolution — they must not

contain any in-built bias in their simulation of the evolutionary process.

An alternative approach to evolution modelling is possible through a more rigorous application of ecological theory. If we create an accurate individual-based ecosystem model, we can then add random mutation to that model, and observe the changing population attributes as the system develops in time. If these implementations are sufficiently accurate and efficient, and simulations can be run of large systems over long time periods, then we can present results that describe the effects produced by mutation and evolution within the systems. We therefore extend our general spatial reaction systems to cover this type of evolutionary simulation, and the resulting reaction-evolution-migration systems (REMS) are directly applicable to ecological studies of species interactions, and show the effects that mutation can produce in such systems. Simulation can show the effects of adaptions to individual behaviour, with reactants switching between competing, co-operating, and parasitic modes of activity, or they can identify the interaction between a single evolving species and some changing environment. However, there may well be other applications for such extended models. In epidemiology, we can currently model a viral epidemic (e.g. AIDS) moving through a simple population through contact infection (see Isham [1988]). It is also possible that the REMS model will also allow the simulation of epidemics of infections that can in some way mutate, and thus change their impact or infectiveness on the population. In biochemistry and botany we could model the effects on reacting systems produced by small adaptions in the nature of the reactants due to slight changes in molecular structure.

1.4 The Role of Parallel Computing

In recent years ecologists have begun to produce computer models of ecosystems that do behave in ways similar to what is observed in nature, i.e. exhibiting spatial and temporal population fluctuations (see Zeigler [1977] and Onstad [1988]). Similarly, in other fields of science, the importance of computer modelling is constantly gaining wider recognition — Kaufmann & Smarr [1993] give a stimulating review of the development of modern techniques. The major obstacle to this advance in the past has been the vast amounts of computer power necessary to run very large models over very long time periods. The advent of parallel computing has changed this by providing massive amounts of power

at reasonable costs, and there is evidence that this new technology is becoming accepted even in the ecological community (see Haefner [1991]) — not traditionally one of the first disciplines to exploit the leading edge in computing hardware.

Ever since computers were first used for serious scientific or applications work, there has been a demand for ever-faster machines. This results from scientists always having larger-scale problems to solve, and always desiring more accurate solutions to current calculations. Since their invention in the 1940s (another important area in which Alan Turing's influence cannot be underestimated — see Hodges [1983]), computers have followed the basic design of a single processor connected to a single memory store, executing one instruction at a time. Until recently the performance of these individual machines has been able to improve through constant innovations in processor technology. However, in the 1990s the ability to improve the performance of single processors has virtually disappeared, although the demand for increased performance remains unceasing. Current processor technology is touching the limits set by the fundamental laws of physics; electrons cannot travel through silicon (or any other substrate) faster than the speed of light; and transistor-type units on microprocessor wafers must remain large enough to avoid quantum effects introducing uncertainty into their operation. The only way to meet fully this increasing computation demand is to abandon dependence upon single processor hardware and look towards parallel processing — the use of many independent processors to solve a single problem. However, such use requires a *decomposition* of our problem into many tasks that can be distributed to the available processing units. In addition, this decomposition must be balanced (in terms of the computation load of each task) to ensure efficient use of the supercomputing resource.

Every major computer manufacturer now recognises that parallel processing, of one kind or another, is where the future of high-performance computing lies. It therefore seems certain that in a few years time the vast majority of all computers in use on large scale numerically intensive work will be based upon a parallel architecture of some kind. Undoubtedly some of these machines, perhaps the majority, will hide the parallelism from the user in some way, and allow "normal" sequential code to be executed. However, leading edge technology and the best achievable performance will almost certainly require the additional effort and expertise of parallel programming by the user.

Parallel computers are currently growing rapidly in acceptance as they enter service first throughout academia (see Wallace [1988]) and more slowly in industry (e.g. Smith *et al* [1991] and Chantler [1993]). The range of current parallel applications goes from common-or-garden database searching for banking institutions (Keane [1993]), to the calculations involved with quantum-chromodynamics in fundamental physics (the UKQCD consortium [1992]). However the power of these machines is not always so easily accessible. Although there are certainly problems that can be transferred to the new generation of supercomputers with very little work, there are also very many that require a serious effort to implement efficiently. This extra work stems from the fact that many problems are, by their nature, difficult to decompose into independent and equally balanced tasks. Because of this, a programmer of parallel computers must often make significant changes to code in order to allow its balanced decomposition. Section 5.2 covers this area in more detail, and lays the ground for the later chapters that discuss the techniques we have developed for efficient parallel implementations of our spatial reaction systems.

1.5 Research Objectives

Since the pioneering work of Turing, relatively little mathematical research has been forth-coming to extend his ideas and methods. This is surprising as he had only just begun his studies in this field before his tragic death, and the techniques he developed have become widespread in their use in many fields of science. Since there has been so little mathematical advancement, scientists seem to have become constrained to use the results produced by Turing's deterministic linearisation techniques. The, admittedly limited, advances in stochastic mathematics never seem to have been fully linked into studies of spatial reaction systems in the applications domain, despite their growing use in more abstract theoretical studies. This is undoubtedly due, in part, to the accuracy, simplicity and applicability of Turing's methods. However we do hope that now is the time to attempt to extend the field, if only by a small amount.

In this work we first give a review of the solution techniques that already exist for linear systems. This involves extending Turing's work to give a complete parameter space specification to identify the regions in which the various forms of linear behaviour exist.

This result is then extended to produce a full classification of the possible behaviour of general spatial reaction systems. This classification is performed in relation to the concept of the systems' *criticality*, i.e. whether populations or concentrations of reactants develop in sub-critical, super-critical or critical patterns. Here we have adopted the terminology from the field of chain reactions, or branching processes. Following this analysis we introduce a mathematical model of a stochastic version of our spatial reaction system. We work through a probabilistic formulation of the linearised equations for a discrete, one-dimensional spatial reaction system, and then solve the resulting equations using a Fourier transform approach. This leads us to an exact measure of the expected behaviour of (linearised) stochastic populations, and these results can be compared to those produced by the linear and non-linear deterministic analyses. This exposes the importance of the use of stochastic versions of real-life systems, if we wish to have any hope of accurately modelling what we observe in reality. In general it is only in stochastic modelling can we reproduce the inherently fluctuating and random attributes of natural systems. We feel that to obtain a true understanding of real-world systems it is necessary for us to enhance our theoretical knowledge of such stochastic systems.

A major part of this work has been the implementation of computer realisations and simulations of deterministic and stochastic systems. Such efforts allow us to confirm the results produced by our analyses, and also to investigate the behaviour of the systems throughout their parameter space. In particular, the ability to view graphical output from deterministic realisations allows us to study transient as well as asymptotic effects. This has proved a vital matter, since it seems that it is often the strength of these transients that determines the long-term state of the system, and as such we can observe unexpected effects within discrete systems that could not be identified by any kind of continuous or discrete analysis. For stochastic models the ability to run system simulations is possibly even more important for obtaining a feel for the way in which the system behaves. In particular, when we move onto the study of evolutionary systems, it is only through stochastic computer simulation that we can provide any results that could show evolutionary mechanisms in action. We therefore present a fairly extensive review of the realisation and simulation results that we have achieved, and their presentation is motivated by successful comparisons between analysis and simulation of both linear and non-linear, stochastic and deterministic systems. In addition, for one particular ecological system we present simulation results that detail the way in which a stochastic

implementation can model realistic system behaviour. Indeed this produces some very interesting insights into the relative stabilities of systems containing reactants with varying attributes. In turn this can lead to the modelling of a complex ecosystem where such differing strains of reactants co-exist.

The third major thrust of this work has been mainly from a theoretical computer science perspective. The massive computational effort necessary to run the simulations that we desire far exceeds that available from standard university mainframe machines, or even from modern dedicated workstations. We have therefore had to turn to supercomputing, and in particular parallel supercomputing, in order to obtain the necessary machine performance. The basis of parallel computing was introduced in Section 1.4 and is more fully explained later in Section 5.2. It constitutes a relatively new area of technological advancement, and as such the techniques to exploit it fully are still under development. These techniques are of general applicability to the modelling of any stochastically varying system, and as such represent a significant advance in certain fields of computer science research (see Smith & Renshaw [1993]).

The work presented here can therefore be seen to cover a fairly broad field of interests. We have attempted to draw the areas together where possible. In particular, we feel that we have established an important link between analytic mathematics and theoretical computer science, in terms of using system parameters to predict the necessary implementation parameters to obtain optimum simulation efficiency. This new ability allows us to make a major investigation in the field of ecological modelling, and it is from this field that the majority of simulations examples are taken. This bias does not only stem from a personal interest in this particular area, but also because it provides such excellent examples of all types of system behaviour, and in particular is by far the best (and possibly the only) source of examples of evolutionary effects.

2

Mathematical Analysis of Deterministic Spatial Reaction Systems

Just as deduction should be supplemented by intuition, so the impulse to progressive generalisation must be tempered and balanced by respect and love for colourful detail.

– Richard Courant

2.1 Introduction to Spatial Reaction Systems

The first step towards a full mathematical understanding of the equations governing spatial reaction systems is the analysis of the deterministic system. Although we should generally consider the case of potentially non-linear interactions (see Section 2.4), we can often make assumptions that simplify the systems (e.g. linearisation) to obtain an approximate solution. The validity of such simplifications can then be quantified analytically, through study of resulting expressions, and also qualified through numerical simulation of both complete and approximate systems (see Chapter 3).

The spatial systems under study in this work have applications in many fields of science, each potentially requiring descriptive equations of differing complexity and dimensionality. We therefore present a general representation of such systems, allowing the subsequent analytic results to be widely applicable. The systems we consider consist of a set of N_c reactants, populating locations organised as a regular grid of dimensionality D with periodic boundary conditions. The latter allows us to avoid edge or boundary effects, and thus produce an approximation to an infinite system. Let us denote the amount of a reactant of type c in location r within dimension k (with $r = 1, \ldots, N_k$ and $k = 1, \ldots, D$) as $X^c_{r_k}$. For systems of competing chemicals $X^c_{r_k}$ will represent a value for the local concentration of a certain chemical, whereas in a population dynamics scenario it may represent the number of individuals of a certain species present in location r_k. The development of such systems in time can be described by a set of N_c coupled ordinary differential equations, thus:

$$\frac{dX^c_{r_k}}{dt} = \mathcal{F}^c(X^1_{r_k}, \ldots, X^{N_c}_{r_k}) + \mu^c(-2DX^c_{r_k} + \sum_{j=1}^{D} X^c_{r_j \pm 1}) \quad \text{for} \quad c = 1, \ldots, N_c \quad (2.1)$$

where $X^c_{r_k \pm 1}$ represents the two terms corresponding to reactant concentrations in sites neighbouring location r in dimension k. This migration term can be considered as a specification of *resultant net flow* regulated by the reactant-type specific migration rate μ^c. We also have N_c functions $\mathcal{F}^c$ that describe the nature of the interaction between reactants within a single location.

In order to enable the analysis of such systems, we will typically use three mechanisms to simplify matters, although consideration is later made of cases devoid of these approximations. First, we generally consider systems containing two reacting agents

($N_c = 2$). In this case we replace the representations for reactant population and migration rate with $X^1 \Rightarrow X, X^2 \Rightarrow Y, \mu^1 \Rightarrow \mu$ and $\mu^2 \Rightarrow \nu$. Second, we will often reduce the dimensionality of the problem to $D = 1$, besides simplifying notation this also allows us to reduce the migration terms to

$$\mu(X_{r-1} - 2X_r + X_{r+1}) \quad \text{and} \quad \nu(Y_{r-1} - Y_r + Y_{r+1}) \ .$$

However, the most significant simplification to the system comes from the linearisation of the interaction functions $\mathcal{F}^c$. In this case we regard all terms higher than first-order to be negligible, whether these are single reactant terms (e.g. terms of $O(X^p)$, where $p > 1$) or cross-interaction terms such as $O(XY)$. In the general case of N_c reactant types we can write the linearised local interaction functions within site r as

$$f^c(X_r^1, \ldots, X_r^{N_c}) = \sum_{j=1}^{N_c} a_j^c X_r^j \quad \text{for} \quad c = 1, \ldots, N_c \ .$$

This introduces the concept of a coefficient of interaction (a_j^c) that describes the weighting attributed to the population of each type of reactant in the linearised interaction function. There thus exist N_c^2 such coefficients which can be divided into two classes — *self-interaction* coefficients when $j = c$, and *cross-interaction* coefficients when $j \neq c$. The cross-interaction coefficients thus describe the nature and strength of the coupling between the differential equations.

2.2 Linearised One-Dimensional Systems

Let us consider the simplified case of spatial reaction systems comprising two reactants in one periodic dimension (such that we have ring of N sites with site 0 indeterminate from site N, and likewise for sites 1 and $N + 1$ etc.), undergoing local interactions defined by the general non-linear functions $\mathcal{F}$ and $\mathcal{G}$. We can therefore write

$$dX_r/dt = \mathcal{F}(X_r, Y_r) + \mu(X_{r+1} - 2X_r + X_{r-1}) \quad \text{and}$$

$$dY_r/dt = \mathcal{G}(X_r, Y_r) + \nu(Y_{r+1} - 2Y_r + Y_{r-1}) \ .$$

Since we are generally interested in using these equations to describe real physical, chemical or biological systems, let us postulate the existence of a stable equilibrium state with real and positive reactant concentration values, X^* and Y^*, such that we have $\mathcal{F}(X^*, Y^*) = 0$ and $\mathcal{G}(X^*, Y^*) = 0$. If we assume the approximation that $X_r(t)$ and $Y_r(t)$ vary around these equilibrium values by only small amounts, $x_r(t)$ and $y_r(t)$, such that

$$X_r(t) = X^* + x_r(t) \quad \text{and} \quad Y_r(t) = Y^* + y_r(t) \,, \tag{2.2}$$

then we can approximate the system by using Taylor's expansion. Ignoring terms higher than first-order in x_r and y_r we obtain

$$\mathcal{F}(X^* + x_r, Y^* + y_r) \simeq \mathcal{F}(X^*, Y^*) + x_r \partial \mathcal{F}(X^*, Y^*)/\partial x_r + y_r \partial \mathcal{F}(X^*, Y^*)/\partial y_r$$

$$\mathcal{G}(X^* + x_r, Y^* + y_r) \simeq \mathcal{G}(X^*, Y^*) + x_r \partial \mathcal{G}(X^*, Y^*)/\partial x_r + y_r \partial \mathcal{G}(X^*, Y^*)/\partial y_r \,.$$

Hence the linearised equations reduce to

$$dx_r/dt = ax_r + by_r + \mu(x_{r+1} - 2x_r + x_{r-1}) \tag{2.3}$$

$$dy_r/dt = cx_r + dy_r + \nu(y_{r+1} - 2y_r + y_{r-1}) \tag{2.4}$$

where

$$a = \frac{\partial \mathcal{F}(X^*, Y^*)}{\partial x_r}, \quad b = \frac{\partial \mathcal{F}(X^*, Y^*)}{\partial y_r}, \quad c = \frac{\partial \mathcal{G}(X^*, Y^*)}{\partial x_r} \quad \text{and} \quad d = \frac{\partial \mathcal{G}(X^*, Y^*)}{\partial y_r}.$$

We have therefore specified the linearised self-interaction coefficients as a and d, and the cross-interaction coefficients as b and c, for our one-dimensional, two-reactant system.

2.2.1 Turing's Method of Solution

Let us study the method introduced by Turing [1952] to solve the linearised equations (2.3) and (2.4). His approach involves taking Fourier transformations of x_r and y_r:

$$u_r = (1/N) \sum_{s=1}^{N} x_s \exp(-2\pi irs/N) \quad \text{and} \quad v_r = (1/N) \sum_{s=1}^{N} y_s \exp(-2\pi irs/N) \,,$$

with standard inverse transformations

$$x_r = \sum_{s=0}^{N-1} u_s \exp(2\pi irs/N) \quad \text{and} \quad y_r = \sum_{s=0}^{N-1} v_s \exp(2\pi irs/N) \ . \tag{2.5}$$

These values can be substituted into (2.3) and (2.4), and making use of the relation

$$\sum_{r=1}^{N} \exp\left[\frac{2\pi irs}{N}\right] = \begin{cases} 0 & \text{if } 0 < r < N \\ N & \text{if } r = 0 \text{or} N \end{cases}$$

we obtain

$$du_s/dt = [a - 4\mu \sin^2(\pi s/N)]u_s + bv_s \quad \text{and}$$

$$dv_s/dt = [d - 4\nu \sin^2(\pi s/N)]v_s + cu_s \ . \tag{2.6}$$

Thus (2.3) and (2.4) been converted into a standard form, with solution

$$u_s = A_s e^{p_s t} + B_s e^{p'_s t} \quad \text{and} \quad v_s = C_s e^{p_s t} + D_s e^{p'_s t}.$$

Here p_s and p'_s are the roots of the equation

$$\left(p - a + 4\mu \sin^2(\frac{\pi s}{N})\right)\left(p - d + 4\nu \sin^2(\frac{\pi s}{N})\right) = bc \ , \tag{2.7}$$

and the constants A_s, B_s, C_s and D_s, determined from initial conditions, satisfy the relations

$$A_s[p_s - a + 4\mu \sin^2(\pi s/N)] = bC_s \quad \text{and} \quad B_s[p'_s - a + 4\mu \sin^2(\pi s/N)] = bD_s \ .$$

These results can now be substituted back into (2.5). Whence replacing the variables x_r and y_r with the actual values X_r and Y_r according to (2.2) Turing obtains the solution

$$X_r = X^* + \sum_{s=1}^{N} [A_s e^{p_s t} + B_s e^{p'_s t}] e^{2\pi irs/N} \tag{2.8}$$

$$Y_r = Y^* + \sum_{s=1}^{N} [C_s e^{p_s t} + D_s e^{p'_s t}] e^{2\pi irs/N}. \tag{2.9}$$

2.2.2 Asymptotic Behaviour of the Linear Solution

For the general linear solution (2.8) and (2.9) to be valid, initial departures from equilibrium around the ring are defined to be sufficiently small such that the linear approximations to the functions $\mathcal{F}(X_r, Y_r)$ and $\mathcal{G}(X_r, Y_r)$ can be used. Although such approximations may be highly inaccurate as populations diverge in *realistic* scenarios, it is often the initial small-scale movements that determine the final state of our full non-linear systems. Therefore, for now, we will assume local linearity in the interactions between X_r and Y_r and study the types of behaviour that can be produced using Turing's solutions. There are two reasons for pursuing this line of study. Firstly, we are interested in initial departures from equilibrium, which by the nature of the linear approximation should be modelled accurately. Secondly, it has been shown by Renshaw [1991] that such approximations can actually perform with remarkable accuracy, even for systems that have moved a great distance from equilibrium.

After an initial perturbation of the equilibrium ring system, there will be a transient phase as the effect of this perturbation is transmitted through the system. In general we are more interested in long term (asymptotic) behaviour, although in some unstable scenarios the transient phase can be seen to dominate the solution — this has often lead to a disregard for these techniques in the past (see Bard & Lauder [1974]). The asymptotic solution to Turing's system is in general a set of spatial *wave*-like variations in cell populations, defined by the $e^{2\pi irs/N}$ terms in (2.8) and (2.9), and brought about by differences in the migration rates driving instability (Murray [1989]). These terms each produce s population *peaks* around the N sites, although in general only one of the N solutions will dominate in the long-term. The scale of this asymptotically dominant feature is in turn governed by the e^{pt} terms in (2.8) and (2.9) as t becomes large. A subset of such situations has been analysed for a certain set of parameters by Turing [1952], and we will give later a complete specification of this parameter space. Alternatively, as we will see in Section 2.3.5 and Chapter 3, in certain circumstances the transient activity of the non-linear system can in fact govern the ultimate system morphology. For the moment, however, let us consider that there are basically six types of linear behaviour associated with the e^{pt} terms. Dependent upon the nature of p, these can be organised into two categories:

- If p is real then e^{pt} behaves in a stationary manner, dependent on the sign of p. As

$t \rightarrow \infty$ then

  - if p is positive, e^{pt} grows indefinitely large,
  - if p is negative, e^{pt} decays to zero, and
  - if p is zero, e^{pt} remains constant at unity.

- If p is complex then e^{pt} oscillates. The amplitude of these oscillations is dependent upon the real part of p. Thus
  - if $\Re(p) = 0$ then the amplitude is constant,
  - if $\Re(p) > 0$ then the amplitude increases, and
  - if $\Re(p) < 0$ then the amplitude decreases.

The persistent solution of our linearised one-dimensional spatial reaction system is clearly dominated, as $t \rightarrow \infty$, by those e^{pt} terms in (2.8) and (2.9) that have real parts that are largest in magnitude; let us denote them by $p = p_{s_0}$. Whether these terms are real or complex then determines the temporal nature (which we term *kilter*) of the solutions. If p_{s_0} is real then we obtain a *temporally stationary* situation with s_0 waves arranged around the ring of cells. These waves consist of a spatial variation in cell populations that is morphologically stable in time. Figure 2.1 shows examples of this behaviour taken from the computer realisation results detailed in Chapter 3. Alternatively, if p_{s_0} is complex we obtain the *oscillatory* kilter case with s_0 waves of oscillating amplitude. Here the waves also consist of spatial variations in cell population, though as time progresses the amplitude of these wave structures oscillates (see Figure 2.2, and later Figure 3.2 for more detail).

Within each type of system kilter we believe that it is important to consider the *criticality* of the solution (this concept will be fully detailed later in Section 2.3). Provided there is genuine instability in the linear system, i.e. the real part of p_{s_0} is positive, then wave amplitude will increase exponentially as time progresses, this is *super-critical* behaviour. Alternatively, if $\Re(p_{s_0})$ is negative we obtain the *sub-critical* case of decaying wave amplitudes. At the boundary between the super- and sub-critical domains we obtain *critical* behaviour (i.e. when $\Re(p_{s_0}) = 0$). Although the super- and sub-critical cases cover the vast majority of possible solutions, the critical case can often produce interesting results. In particular, it is at criticality that strong transient

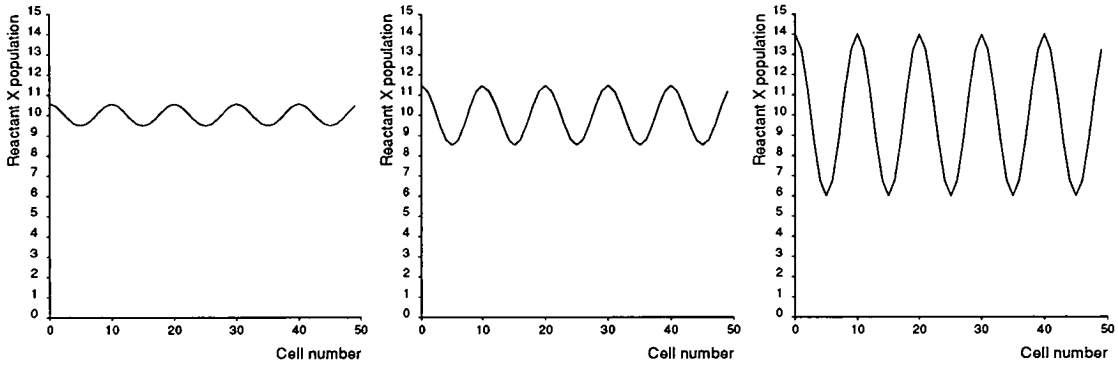

Figure 2.1: Reactant waves in the stationary kilter

Time evolutions of temporally stationary waves in our standard realisation scenario. Using a ring of $N = 50$ cells, migration and interaction coefficients are chosen to produce $s_0 = 5$ and $p_{s_0} = 0.1$. The three graphs represent stationary kilter, linear wave growth at times $t = 10, 20$ and 30.

activity can overwhelm any expected asymptotic behaviour. We thus have three classes of criticality for our linearised system, all of which can exist in both system kilters — stationary and oscillatory. We therefore have a total of six possible modes of system behaviour.

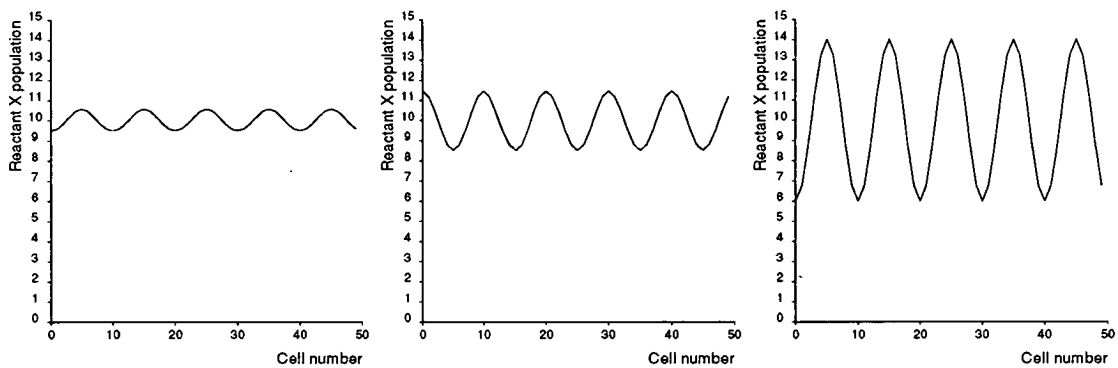

Figure 2.2: Reactant waves in the oscillatory kilter

Time evolutions of temporally oscillatory waves in our standard realisation scenario. Using a ring of $N = 50$ cells, migration and interaction coefficients are chosen to produce $s_0 = 5$ and $p_{s_0} = 0.1$. The three graphs represent oscillating kilter, linear wave growth at times $t = 10, 20$ and 30.

2.2.3 Turing's Behavioural Classification

In his pioneering paper Turing [1952] classifies six possible behavioural sub-cases, all within the super-critical scenario. He concentrates his efforts in this criticality region since it is here that wave structures will be persistent, although under the linearisation

approximation these waves will always have exponentially growing amplitudes. Two of his cases can only be realised if more than two reactants are present, and we will now concentrate on the other four cases when two species X and Y exist. These cover the stationary and oscillatory cases of super-critical activity, and can be summarised as follows:

1. *Stationary case with extreme-long wavelength.*

   This situation will exist if, for example, $\mu = \nu = 1/4$, $b = c = 1$, and $a = d$. These parameters produce p_s values that are always real, and are largest when $s = 0$. The dominant solution will therefore have neighbouring cell populations evolving in close synchronisation, although there will not necessarily be any direct association between distant cells. There is a general equilibrium around the ring, with groups of cells acting in a similar manner due to strong short range correlations between reactant populations. Turing suggests that this effect is unlikely to be very interesting on a ring of cells, and this is supported by the realisation results in Figure 2.3. These show the gradual spread of global effects (there is no differential migration) dependent upon initial perturbations.

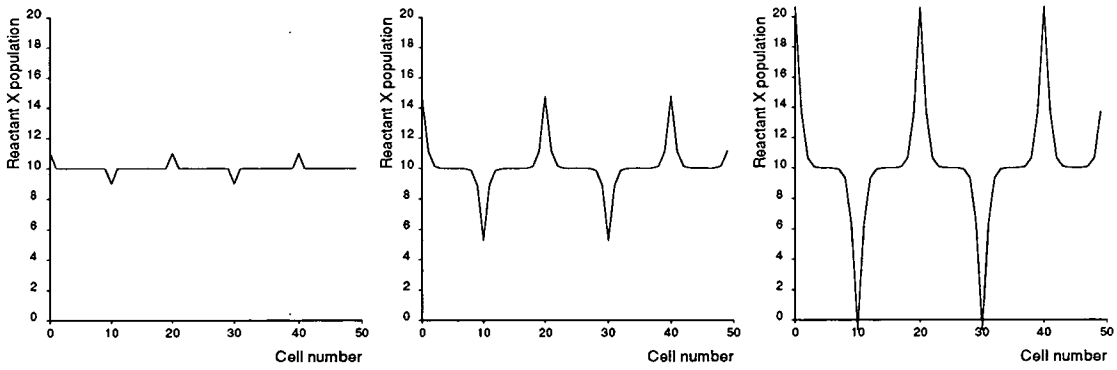

Figure 2.3: Stationary extreme-long wavelength reactant waves
Time evolutions of the first of Turing's wave classifications using 50 sites in one dimension. The three graphs represent growth of X-reactant population waves at times $t = 0$ (showing the initial perturbations), $t = 1$ and $t = 1.5$.

   However this can produce some interesting *dappled* patterns in two dimensions, and therefore proves more interesting for study with a two-dimensional deterministic implementation (Section 3.6) and in the stochastic scenario (Section 5.4).

2. *Oscillatory case with extreme-long wavelength*

   This case will dominate when, for example, $\mu = \nu = 1/4$, $b = -c = 1$, and

$a = d$. This is very similar to the first case; now p_s is complex, but the largest $\Re(p_s)$ still occurs when $s = 0$. There is again strong local correlation, but very weak distant correlation. This could again result in dappling or other interesting behaviour in a two-dimensional or stochastic system. The difference between this and the previous case is that any movement of a cell population away from the equilibrium state will be oscillatory rather than stationary. This effect is shown graphically in Figure 2.4.

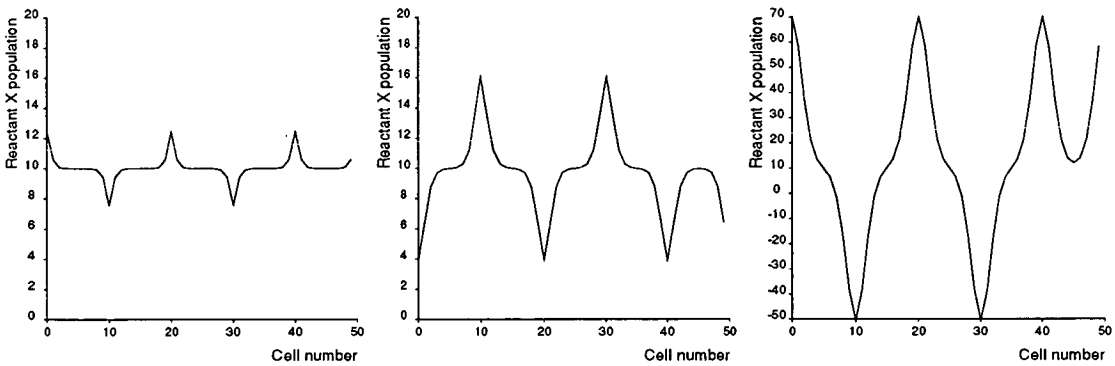

Figure 2.4: Oscillatory extreme-long wavelength reactant waves
Time evolutions of the second of Turing's wave classifications using 50 sites in one dimension. Following the same initial perturbations as in Figure 2.3, these three graphs represent oscillatory growth in X-reactant populations at times $t = 1, 3$ and 6. Note that the final graph uses a different scale.

3. *Stationary case with extreme-short wavelength.*

   An example of this behaviour can be produced using $\mu = 1$, $\nu = 0$, $d = I$, $a = I - 1$ and $b = -c = 1$, where I is Turing's *Instability* parameter. In this scenario p_s is always real and is greatest when $s/N = 1/2$, i.e. each wave exists within just two neighbouring locations. With the ring system in general equilibrium, any perturbations will either decay back to equilibrium (for $I < 0$ — the sub-critical behaviour not considered further by Turing) or explode exponentially (for $I > 0$ — super-critical behaviour). The drift away from equilibrium will tend to be in opposite directions for neighbouring cells, thus producing $N/2$ *waves* around the ring. There is therefore a large correlation between all populations that are multiples of two cells apart. Figure 2.5 shows a graphical realisation result for such a stationary, sub-critical extreme-short wavelength case. This is unlikely to provide many interesting examples of natural behaviour, but what may be interesting is how much of the total parameter space falls within this category.

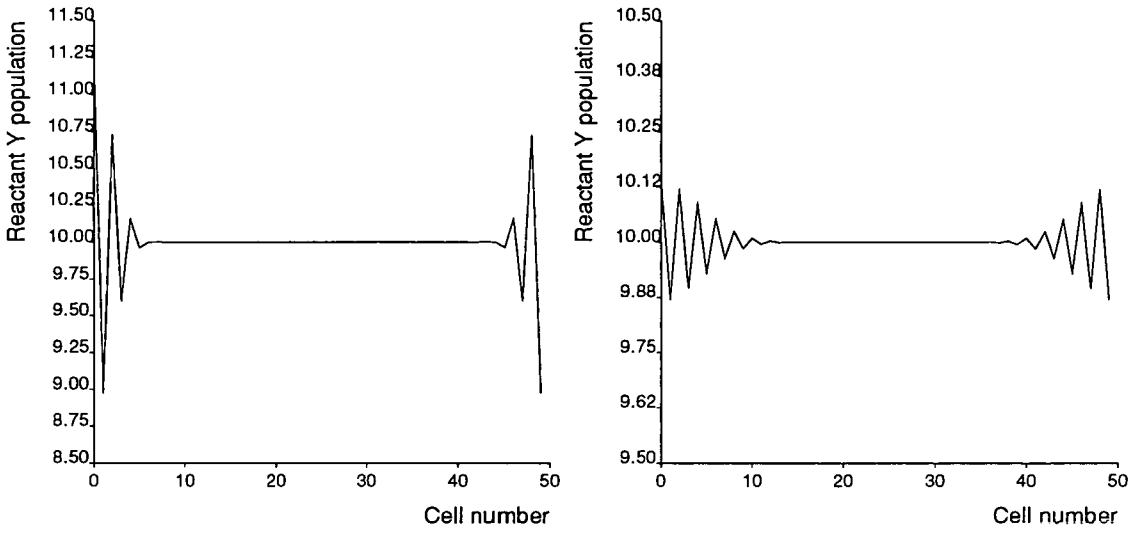

Figure 2.5: Stationary waves of extreme-short wavelength

Snapshots of a two-reactant Turing ring system, using parameters necessary to produce sub-critical stationary waves of extreme-short wavelength. These graphs show Y-reactant populations at times $t = 40$ and $t = 200$ following a single perturbation. The extreme-short wavelength patterns are clearly visible, as is the steady advance of the perturbation effect around the ring of sites. Note the change of scale between the two graphs as the sub-critical wave amplitudes decay.

4. *Stationary waves of finite wavelength.*

   This is the most interesting of Turing's six cases, as it provides a direct mathematical link to the production of stable wave patterns from simple interactions. Turing quotes one set of specific parameters that can produce this category of behaviour as follows:

$$a = I - 2, \quad b = 2.5, \quad c = -1.25, \quad d = I + 1.5, \quad \text{and}$$

$$4\mu \sin^2(\frac{\pi s_0}{N}) = 8\nu \sin^2(\frac{\pi s_0}{N}) = 1/2 \ .$$

   These parameters will produce s_0 symmetric waves around a ring provided that s_0 divides exactly into N; should this not be the case then waves with a wavelength corresponding to the nearest whole integer to N/s_0 will be produced. Turing states that these waves will grow from any perturbation of the super-critical (i.e. $I > 0$) system in equilibrium, although the long-term physical nature of these waves must be questioned because of the exponential growth in their amplitude. In effect we should only regard the general morphology of these systems as representative of physical behaviour, since as soon as cell populations become

large our linearisation approximation ceases to be valid. As we will see in Section 3.2 this problem can disappear for non-linear interactions where amplitude growth is restricted. In addition, in Chapter 3 we will show that the linearised situation is actually much more complex and interesting than Turing suggests, as perturbation size and location can produce transients that dominate asymptotic system activity.

In Turing's analysis of the stationary waves of finite wavelength case he considers only the case of $I > 0$. Therefore all wave scenarios produce explosive exponential growth under the linear approximation. We will now extend Turing's efforts in order to give a complete theoretical analysis of the parameter space for the linearised two-reactant one-dimensional system. This will in turn allow a more thorough experimental analysis of the parameter space, in particular the production of permanent wave patterns, and a study of system stability.

2.3 Criticality in Linear Systems

Our objective in this section is to provide a second level of specification for the behaviour of our standard one-dimensional spatial reaction systems. Much biological study has centred on such models (see Murray [1989] for a review), although little emphasis is placed even on Turing's division of stationary and oscillatory kilter. In the study of morphology it is often simply the size and nature of *pattern* that is of interest. We believe that not only kilter-state, but also system criticality should be considered for all such models, as these can highlight fundamental problems with current models that can actually lead them away from representing true physical or biological systems.

We have found that an understanding of criticality provides a rationale for certain scenarios being dominated by initial conditions or transient activity. In addition, consideration of the oscillatory/stationary kilter divide can expose reasons for difficulty in producing particular behaviours in certain systems. In particular, we show that in some scenarios a fine-grain diffusion model cannot produce large-scale effects without the incorporation of unrealistic parameter values.

2.3.1 Parameter Space Specification

The comparatively straightforward one-dimensional, two-reactant system analysed by Turing contains a wealth of possible parameter sets. In his initial analysis, this is countered by choosing specific values for coefficients in order to produce a sub-set of the various types of system behaviour. This approach is ideal for exposing and demonstrating specific details of possible system activity. However, in this work we are more concerned with exploring the actions of such systems when parameters are chosen to lie close to stability limits. We therefore investigate areas of parameter space in more detail, learning about the structure of regions within the space, and hence enabling a full study of the system behaviour.

It must be noted that despite the simplifications inherent with linearisation, and the consideration of only the one-dimensional system of two reactants, the free variables in the general system solutions still number seven – two self-interaction coefficients (a and d), two cross-interaction coefficients (b and c), two migration rates (μ and ν) and the number of cells in the ring (N). Besides the difficulties involved with the size of such a high-dimensional parameter space, the value of any analytic results will always be tempered by the difficulties presented in attempting to interpret such analyses. Therefore, any reduction in the number or range of free system variables is useful.

We can achieve some simplification by incorporating some of the parameter restrictions detailed by Turing, and we can ascertain others from considerations of the physical nature of the system. This task is simplified if we restrict our consideration to the specific types of system behaviour in which we are particularly interested. We believe this approach to be legitimate as it allows us to obtain a precise specification of scientifically interesting scenarios, albeit at the cost of rigorous detail. In addition, once complete, a specific analysis can be used to describe the complete parameter space by allowing previously fixed parameters to vary, and then detailing the effect this has on the initial results.

In essence, a full description of parameter space equates to a complete understanding of the behaviour of Equation (2.7). Such an understanding will provide us with the general roots of the equation, where the nature of the largest root then determines the system kilter and criticality. Let us therefore rewrite (2.7) in a slightly simplified manner by introducing two new terms: the ratio of reactant migration rates $m = \mu/\nu$; and the system *periodicity*, λ', defined as the ratio s/N. This latter ratio equates to

the reciprocal of the number of ring locations within a wavelength. In our linearised solution this simplification can be achieved without loss of generality, although both N and s must retain discrete integer values for a real physical system. We can thus express (2.7) as

$$(p - a + U)(p - d + U/m) = bc\,, \tag{2.10}$$

where $U = 4\mu \sin^2(\pi\lambda') = 4m\nu \sin^2(\pi\lambda')$. The roots of (2.10) can be determined as

$$p_{1,2} = \frac{a+d}{2} - U\frac{m+1}{2m} \pm \frac{1}{2}\sqrt{4bc + \left(\frac{U}{m}(m-1) + (d-a)\right)^2}\,. \tag{2.11}$$

Let us now concentrate on our region of particular interest. The classifications of asymptotic behaviour detailed in Section 2.2.3 reveal three main categories of system activity in terms of the *wavelength* of the structures apparent in the system. Two of these classes are limiting extremes of system activity: the case of "extreme-long wavelength" occurs when migration parameters are chosen to be equal (i.e. $m = 1$) so as to give $s_0 = 0$, and hence all cells act in an identical manner; and the case of "extreme-short wavelength" where all cells act in total independence, although there is a strong correlation between the populations of next-nearest neighbour locations (see Section 2.2.3). Interesting system activity occurs in the region between these two extremes where observable wave structures dominate activity. These are the cases of "finite wavelength", and it is in this region (equating to the vast majority of parameter space) that we will focus our attention.

This representation does not allow us to reduce the number of free variables in our system. However, it and other physical considerations do place some specific range restrictions on certain parameters:

- $\mu, \nu > 0$ enforces genuine physical migration/diffusion,
- $\mu \geq \nu$ loses no generality as $\nu \geq \mu$ can be generated by exchanging the two reactants, and
- $N > 0$ and $s > 0$ dictates the use of a *bona fide* physical system.

We now introduce additional restrictions and simplifications to remove degrees of freedom that are unimportant to qualitative behaviour. Later in Section 2.3.5 we will

study variation of currently fixed parameters, but for now we are interested in gaining an insight into the details of specific system behaviour. To this end we choose to fix certain coefficients that allow us to simplify the solution of (2.11) and hence determine the leading eigenvalue solutions. We can make use of some of Turing's suggested parameters, as we choose to keep the cross-interaction coefficients (b and c) constant and introduce an *instability* variable (I) that is used to specify a constant relationship between the self-interaction coefficients (a and d). The following analysis is therefore built upon the assumption that $b = -2c$ (ensuring $bc < 0$) with b set to 2.5, the self-interaction coefficients are defined as $a = I - 2$ and $d = I + 1.5$, and the ratio of migration rates $m = \mu/\nu = 2$. The choice of these values leads to the solution roots of (2.11) being given by

$$p_{1,2} = I - \frac{1}{4}(1 + 3U) \pm \frac{1}{4}\sqrt{U^2 + 14U - 1}\,, \tag{2.12}$$

where the value of $p_1 = \max(p_1, p_2)$ determines asymptotic behaviour.

We have therefore reduced the degrees of freedom in this system to three free variables – the system instability (I), migration rate (μ) and periodicity (λ'). We can now identify the regions of this three-dimensional parameter space in which different variants of the finite wavelength scenario occur. The space must be partitioned to reflect system kilter – whether the resulting wave structure is stationary or oscillatory — and for each kilter a criticality partition must be made to describe whether the system is critical, sub-critical or super-critical.

2.3.2 The Stationary–Oscillatory Divide

The identification of the division between oscillatory and stationary waves is straightforward from a consideration of equation (2.12). Oscillatory systems result from the largest eigen-value $\max(p_1, p_2)$ being complex, and this will occur for our system when

$$U^2 + 14U - 1 < 0, \quad \text{i.e. when} \quad -7 - \sqrt{50} < 4\mu \sin^2(\pi\lambda') < -7 + \sqrt{50}.$$

The system kilter divide is therefore independent of instability, I, and since we can ignore all negative μ values, we can plot the division line in (μ, λ')-space where $4\mu \sin^2(\pi\lambda') =$

$\sqrt{50} - 7$ (see Figure 2.6). This gives the separation of the two main behaviours in terms of the two relevant parameters, μ and λ'.

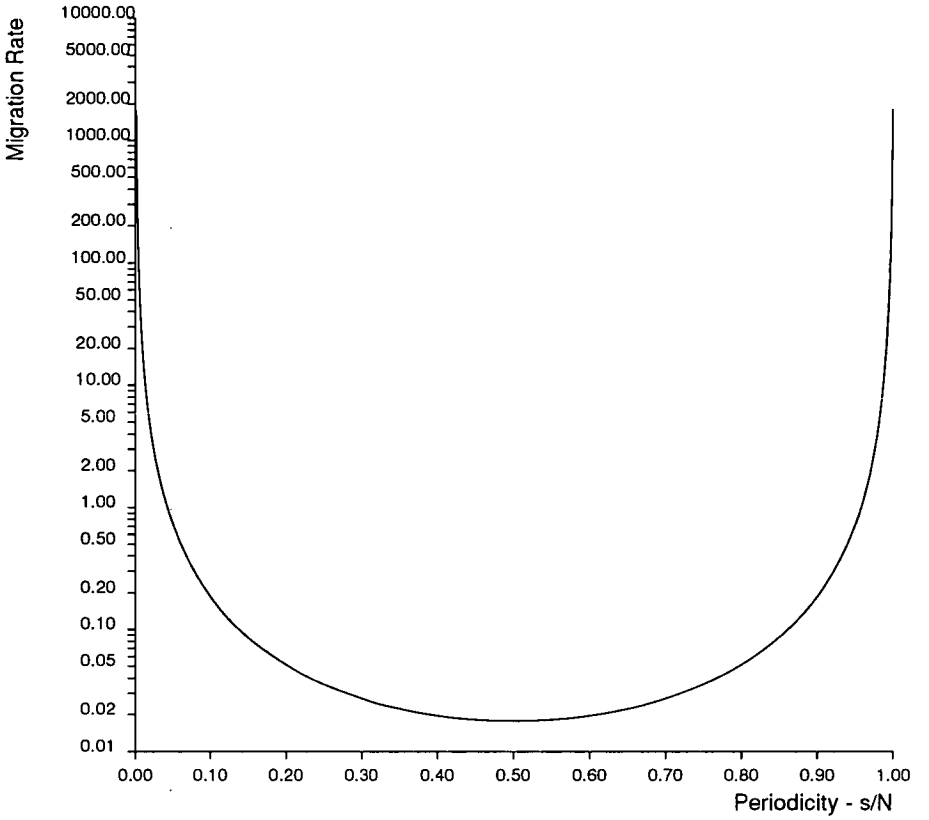

Figure 2.6: The stationary/oscillatory divide

Locus of points lying on the parameter space boundary between stationary and oscillatory wave structures in the linearised, one-dimensional two-reactant Turing system. Given fixed values of $b = -2c = 2.5$, $m = 2$, $a = I - 2$ and $d = I + 1.5$, the result is independent of I and specifies the section of (μ, λ')-space for oscillatory waves as below the curve, and that for stationary waves as above the curve. Note that the migration parameter axis uses a log scale.

The division presented in Figure 2.6 highlights some important features of this kilter divide. First, note that the curve is reflected about the line $\lambda' = 0.5$. At this point we have reached one of our limiting cases – that of extreme-short wavelength. Thus all points for $\lambda' > 0.5$ are unimportant for the real system, since we are left with less than two cells per wavelength (and hence an un-realisable physical system). However, the full curve is shown here for the sake of analytic completeness. We can also observe the other limiting case behaviour (extreme-long wavelength) as $\lambda' \to 0$. As we approach this point we see that the system will be oscillatory unless μ is exceptionally large. Indeed, when $\lambda' = 0$ we would require μ to be infinite to obtain stationary waves of finite wavelength: as we increase the size of our particular system, while holding the

number of waves constant, we must use very large migration rates to ensure temporally stationary rather than oscillatory waves. For example, to obtain five stationary waves on 1000 sites, the migration rate must be in excess of 711. This problem highlights the difficulty that diffusion-based or fine-grain models can have in producing large scale (relative to the size of individual sites) features in certain scenarios.

The oscillatory/stationary division curve also allows us to specify the lower bound on the migration rate necessary to ensure stationary waves for all possible periodicities. We can see from Figure 2.6 that for our standard parameter set, the overall minimum migration rate for stationary behaviour occurs when $\lambda' = 0.5$ and equates to $\mu_{min} = 0.0178$. For an example of a typical scenario used in our simulation work, if one considers using a specific periodicity (e.g. $\lambda' = 0.1$), then for stationary waves we must use a migration rate $\mu \geq 0.1861$. Conversely, given a particular migration rate, Figure 2.6 allows us to define the set of possible wavelengths that could be achieved on a system of given size.

Having successfully defined the division between these two main classes of "finite wavelength" kilter, let us now move on to consider how to subdivide each of these into specific classes of criticality.

2.3.3 Criticality in the Oscillatory System

Given that we have an oscillatory wave structure in our system, i.e. we have chosen system parameters within (μ, λ')-space that lie below the curve in Figure 2.6, we can now define the parameter regions that correspond to critical, super-critical and sub-critical behaviour. From Section 2.2.2 we know that criticality depends upon the sign of the real part of the largest eigenvalue solution (p_{s_0}), and manifests itself in the evolution of the physical system in terms of wave amplitude development. In the sub-critical case, with $\Re(p_{s_0}) < 0$, we will observe long-term wave structures with amplitudes that decrease with time at an exponential decay rate, thereby asymptotically approaching equilibrium. When $\Re(p_{s_0}) > 0$ we have the super-critical case, where any waves that are not due to initial transients will have exponentially increasing amplitudes. Lastly, we have the critical case when $\Re(p_{s_0}) = 0$. Here long-term wave structures will have a constant amplitude; although such a system would be stable in equilibrium, a perturbation may lead to transient structures strong enough to determine the long-term state of the system,

since there are no asymptotically dominant solutions to outweigh the transient activity.

We can observe from (2.12) that the real part of the largest eigen-value of the solution for the oscillatory case is

$$p_{s_0} = I - (1 + 3U)/4 \ .$$

We must therefore consider our third global parameter, the system instability (I), in order to establish system criticality. For our partially constrained system this is achieved by considering the surface in (I, μ, λ')-space that represents all points such that

$$I = \frac{1}{4}(1 + 3U) = \frac{1}{4}(1 + 12\mu \sin^2(\pi\lambda')) \ . \tag{2.13}$$

This surface is shown graphically in Figure 2.7, and represents the set of (I, μ, λ')-values that correspond to critical behaviour. It therefore provides a division between the parameter regions that represent super- and sub-critical activity, the former being the volume above the surface, and the latter being that below.

The main feature of interest from Equation (2.13) and Figure 2.7 is that, in the oscillatory kilter, we will always observe sub-critical, decaying amplitude waves for instability values $I < 0.25$. The minimum instability value to achieve criticality is therefore $I_{min} = 0.25$, although this only occurs in the limit of extreme-long wavelength ($\lambda' = 0$) or extreme-short wavelength ($\mu = 0$). The value of I_{min} is determined purely by the relation between the self-interaction coefficients a and d. For finite wavelength classes I_{min} must increase as either migration rate or periodicity increase. Although Figure 2.7 gives migration rates only up to $\mu = 10$, the rising trend in the bell-shaped surface continues linearly with μ. However, if we recall that not all of the (μ, λ')-space represents inherent oscillatory behaviour, we must reconsider the results in Section 2.3.2 for a specification of this division, and incorporate the resulting structure shown by Figure 2.6 in our representation. This coupling leads us to Figure 2.8 in which we have added a third dimension to Figure 2.6, and super-imposed this upon the surface detailed in Figure 2.7. Thus Figure 2.8 is a union of the two previous diagrams (2.6 and 2.7), and highlights the region of the criticality surface that is valid for the case of oscillatory waves. We see (from the colour coding) that, for the oscillatory wave case, the criticality surface is always very close to $I = 0.25$ for our particular choice of parameter restrictions.

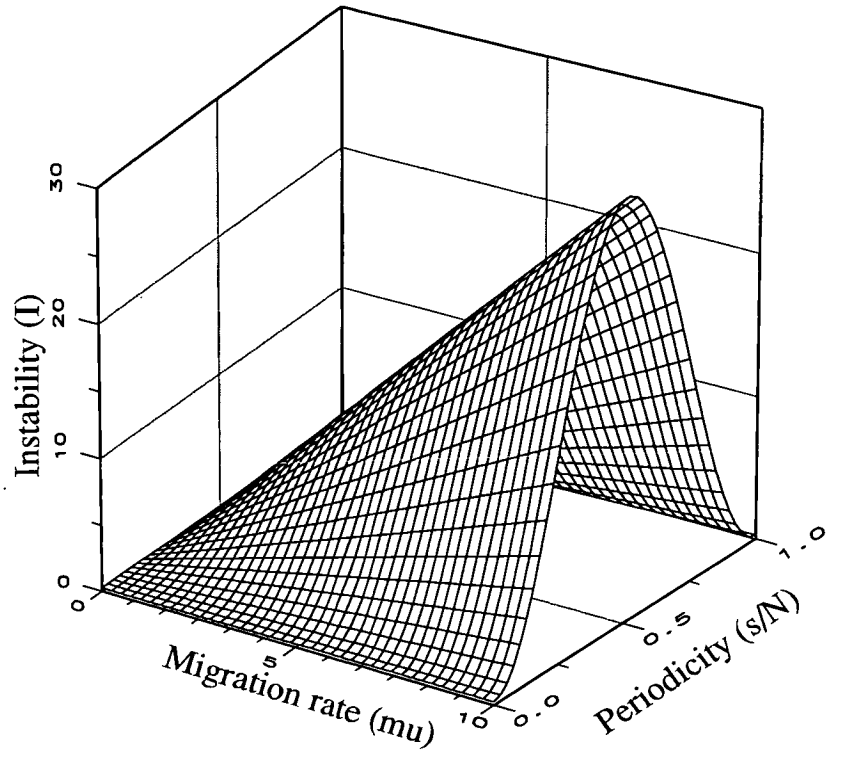

Figure 2.7: The criticality boundary in the oscillatory domain
The criticality boundary for the oscillatory wave structure scenario in the linearised, one-dimensional two-reactant Turing system. Given fixed system parameter values of $b = -2c = 2.5$, $m = 2$, $a = I - 2$ and $d = I + 1.5$, the figure shows the inter-dependence between I, μ and λ' necessary to produce the three criticality cases. The region below the surface corresponds to sub-critical behaviour, and that above to super-critical behaviour. Critical behaviour is represented by points on the surface itself.

2.3.4 Criticality in the Stationary System

Let us now consider the three criticality regions within the domain of stationary waves. In this case we know that the largest eigen-value of the solution of (2.12) is real and can be expressed as

$$p_{s_0} = I - \frac{1}{4}(1 + 3U) + \frac{1}{4}\sqrt{U^2 + 14U - 1} \ .$$

We can now proceed as in Section 2.3.3 and describe the parameter region for critical behaviour as the surface where $p_{s_0} = 0$, i.e. where

$$I = \frac{1}{4}(1 + 3U) - \frac{1}{4}\sqrt{U^2 + 14U - 1}. \tag{2.14}$$

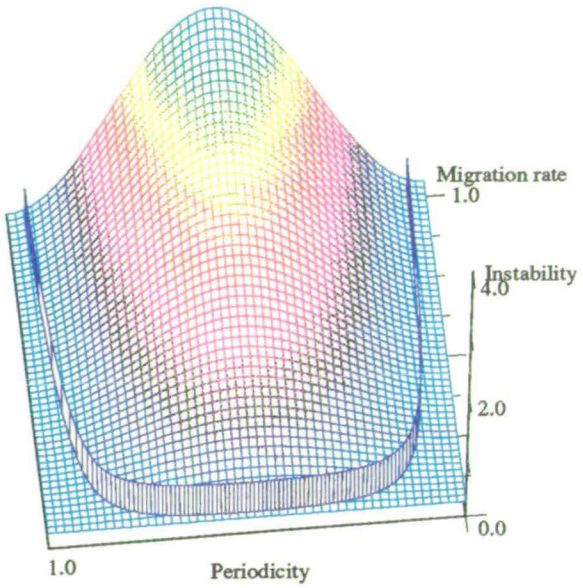

Figure 2.8: A union of criticality and kilter divisions

Definition of the region of criticality that is valid for the oscillatory wave case in the linearised, one-dimensional two-reactant Turing system. Given fixed values of $b = -2c = 2.5$, $m = 2$, $a = I - 2$ and $d = I + 1.5$, the dark blue barrier represents the division of oscillatory from stationary behaviour (as in Figure 2.6), and thus shows that only a small part of the criticality surface is relevant to this scenario. The colour shading used in the diagram is used to convey the height of the criticality surface, and is based on a *rainbow* colour-map assigned linearly between the maximum and minimum height values. This shows that the criticality value of I is close to 0.25 throughout the oscillatory domain.

From (2.14) it is clear that the surface will be lower in the I-dimension than that for the oscillatory case (Figure 2.7), although any comparison between the two is purely notional, as they are each only valid on their respective sides of the oscillatory/stationary divide. However, for the sake of comparison, Figure 2.9 is provided to expose this variation.

Figure 2.9 reveals some interesting features of the stationary criticality surface. We see that in general, as μ and λ' increase, we again need increasing values of instability (I) to obtain critical behaviour. The diagram also shows that the surface rises in the I-dimension as (μ, λ')-values approach the oscillatory/stationary divide. This can be better viewed in a complete *impulse* plot of criticality values shown in Figure 2.10. Here,

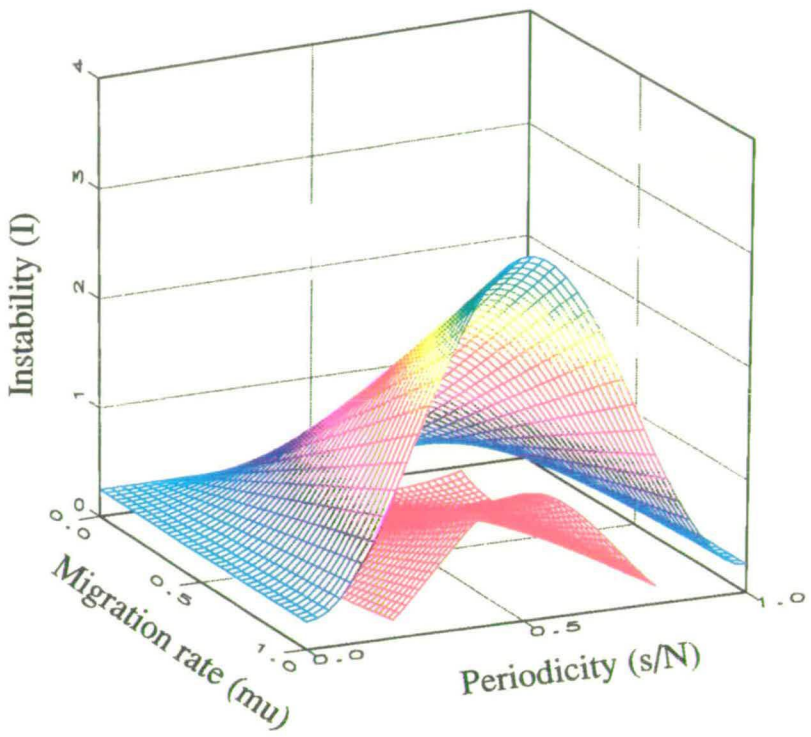

Figure 2.9: Comparison of oscillatory and stationary criticality surfaces
A notional comparison of the criticality surfaces for the oscillatory and stationary scenarios in the linearised, one-dimensional two-reactant Turing system. Given fixed values of $b = -2c = 2.5$, $m = 2$, $a = I - 2$ and $d = I + 1.5$, the red mesh corresponds to the stationary case criticality surface, and the coloured mesh to the oscillatory case. The objective here is to highlight the effects of the extra terms in the definition of the surface.

as (μ, λ')-values approach the division between the stationary and oscillatory domains, the value of I necessary to achieve criticality (I_c) approaches that for the oscillatory case (i.e. for our standard system $I_c \simeq 0.25$). However, as μ or λ' increase from this boundary, I_c initially decreases to a minimum value of (I_0), before increasing again (considering $\lambda' \leq 0.5$). We can identify the locus of I_0-values within the stationary domain, since it must occur when

$$\frac{1}{4}(1 + 3U) = \frac{1}{4}\sqrt{U^2 + 14U - 1} \ .$$

This can be simplified to $U = 1/2$, and thus occurs when $\mu = 1/8 \sin^2(\pi\lambda')$.

Figure 2.11 shows the relative positions of the oscillatory/stationary division and the

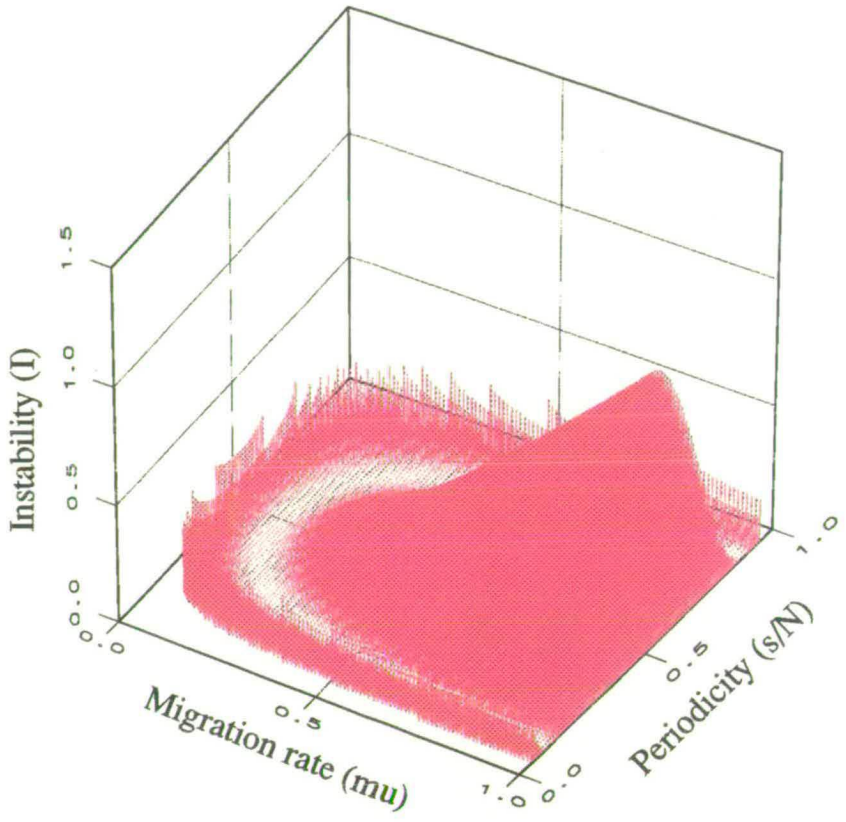

Figure 2.10: The criticality surface in the stationary domain
An impulse plot of the criticality values for the stationary wave domain in the linearised, one-dimensional two-reactant Turing system. Given fixed values of $b = -2c = 2.5$, $m = 2$, $a = I - 2$ and $d = I + 1.5$, the points that correspond to the dividing surface between sub-critical behaviour (below the points) and super-critical behaviour (above the points) are plotted as *impulse* lines (i.e. vertical lines are drawn from each point down to the (μ, λ')-surface).

locus of migration rate and periodicity pairs that allow stationary criticality with minimum instability. By identifying a point upon the periodicity curve we can select (μ, λ')-values to give stationary wave structures at criticality with minimum system instability. For example, with a system periodicity of $\lambda' = 0.1$, we can use a migration rate of $\mu = 1.309$ to obtain criticality with $I_0 = 0$.

2.3.5 A Unified Kilter and Criticality Description

For our particular case of interest, in the linearised one-dimensional two-reactant Turing system it is possible to unify the descriptions of system kilter and criticality that have

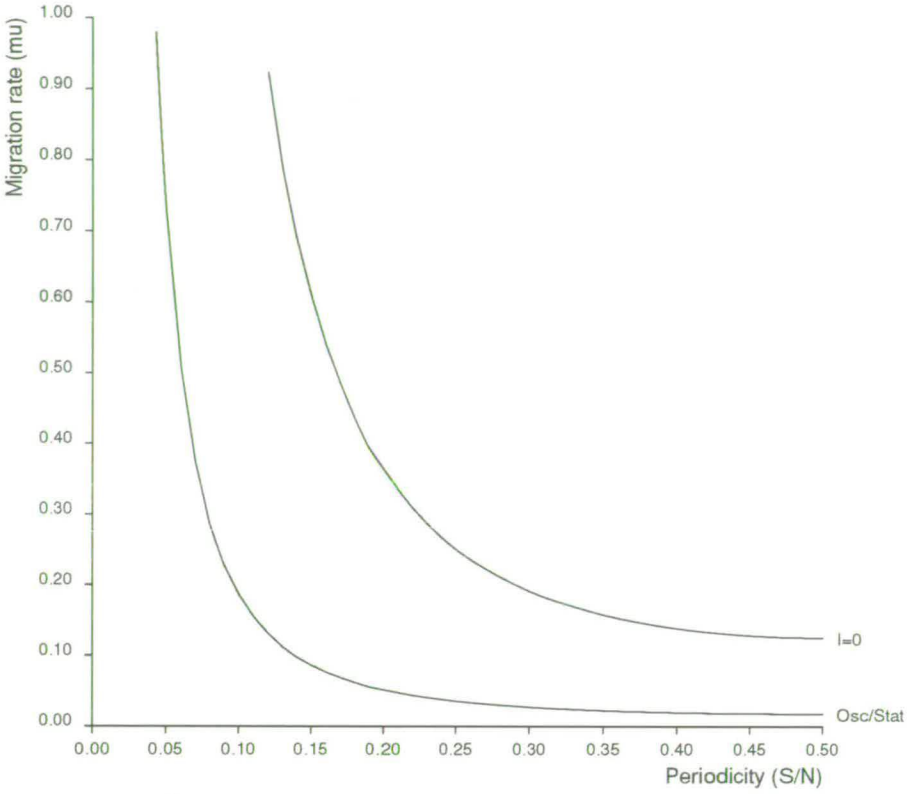

Figure 2.11: Kilter divide and minimum instability positions

A comparison of the loci of oscillatory/stationary division and minimum instability to give stationary criticality in the linearised, one-dimensional two-reactant Turing system. The following fixed values are used: $b = -2c = 2.5$, $m = 2$, $a = I - 2$ and $d = I + 1.5$.

been detailed in the previous sections. We can achieve this with a surface plot of I_c values for the complete (μ, λ')-parameter space (see Figure 2.12). This surface thus acts as a complete description of the (I, μ, λ')-values that produce critical behaviour, and is possible because at the interface between possible system kilters the critical instability values converge.

The surface shown in Figure 2.12 clearly shows the divide between oscillatory and stationary domains as a distinct U-shaped ridge, and also highlights the I_0 *valley* between the kilter divide and the eventual rise in I_c with both migration rate and periodicity. Given this surface, we can clearly identify system kilter and criticality for any given set of parameters.

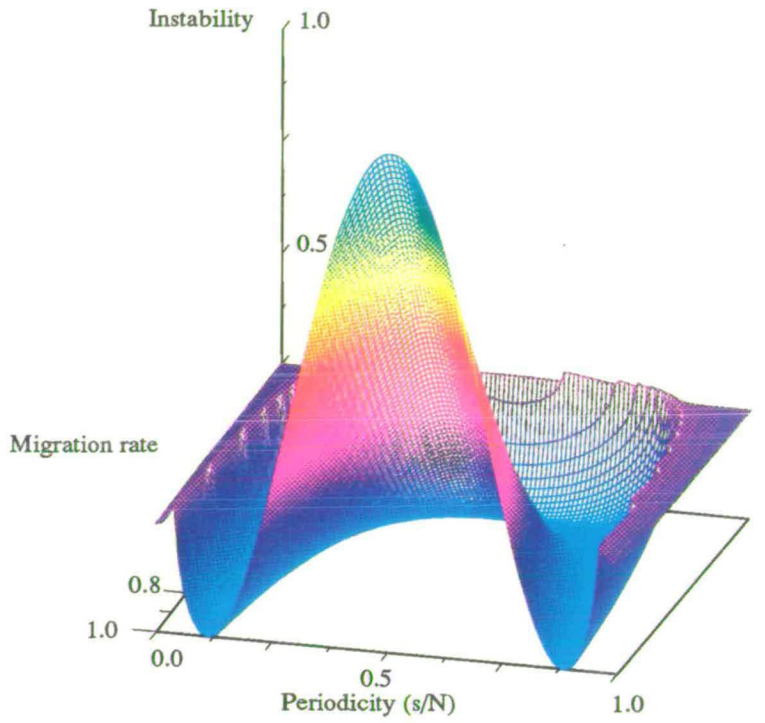

Figure 2.12: Unified criticality surface in linear parameter space
The unified criticality surface for the linearised one-dimensional two-reactant Turing system. Given the fixed values $b = -2c = 2.5$, $m = 2$, $a = I - 2$ and $d = I + 1.5$, this surface defines the regions of system kilter as being either side of the distinct purple ridge, with oscillatory behaviour lying axes-side of this divide. It also defines the regions of criticality (on the surface), sub-criticality (below the surface) and super-criticality (above the surface).

An Extended Kilter Specification

Having reached a unified criticality specification, let us now re-generalise our system to demonstrate the variation of the criticality surface with the (currently) fixed interaction parameters a, b, c, d and the ratio of migration rates, m. Let us reconsider equation (2.11), abandoning use of the simplifying instability parameter, I, and retaining the self-interaction coefficients a and d. For this more complete description of our system we see that we obtain oscillatory behaviour when, from (2.11),

$$4bc + (U(m-1)/m + d - a)^2 < 0 \quad .$$

This allows us to define the locus of points lying on the stationary/oscillatory divide as

$$(U(m-1)/m + d - a)^2 = -4bc \ . \tag{2.15}$$

It must be noted that if $bc \geq 0$ then we always obtain stationary behaviour. To allow the possibility of oscillatory motion (a necessity if we intend to describe the divide between the two kilters) we must therefore have $bc < 0$ (as used earlier). We find the values of U (a function of migration rate magnitude and system periodicity) that satisfy (2.15) to be

$$U_{1,2} = \frac{m}{m-1}(a - d \pm 2\sqrt{-bc}). \tag{2.16}$$

This allows us to specify the region of oscillatory kilter as being when

$$\frac{m}{m-1}(a - d - 2\sqrt{-bc}) < 4\mu \sin^2(\pi\lambda') < \frac{m}{m-1}(a - d + 2\sqrt{-bc}) \ .$$

We therefore see that the restriction $bc < 0$ also allows us to deal exclusively in the real domain. In addition, if we consider $bc = 0$, we see that the two roots of (2.15) converge to

$$U_{1,2} = m(a-d)/(m-1) \ ,$$

which again produces only stationary behaviour since the oscillatory region between the roots has shrunk to zero. In general, therefore, we will consider all values of $bc < 0$. It is obvious from (2.16) that as $|bc|$ grows, so does the oscillatory region of parameter space which is bounded by the $U_{1,2}$ values.

The most fully representative case for kilter division occurs when both the lower and upper bound on U lie within the physically realistic region of parameter space, i.e. $U = 4\mu \sin^2(\pi\lambda') > 0$, with $\mu \geq 0$ and $0 < \lambda' < 0.5$. Since we can restrict the migration ratio to $m \geq 1$ (as $m < 1$ can be accounted for by a reversal of reactants) we can specify the nature of the lower and upper bands according to the values of a and d, the two remaining free parameters, as follows:

- If $a = d$ then
  - lower bound, $U_1 = -2m\sqrt{-bc}/(m-1)$ is always negative, and
  - upper bound, $U_2 = 2m\sqrt{-bc}/(m-1)$ is always positive.

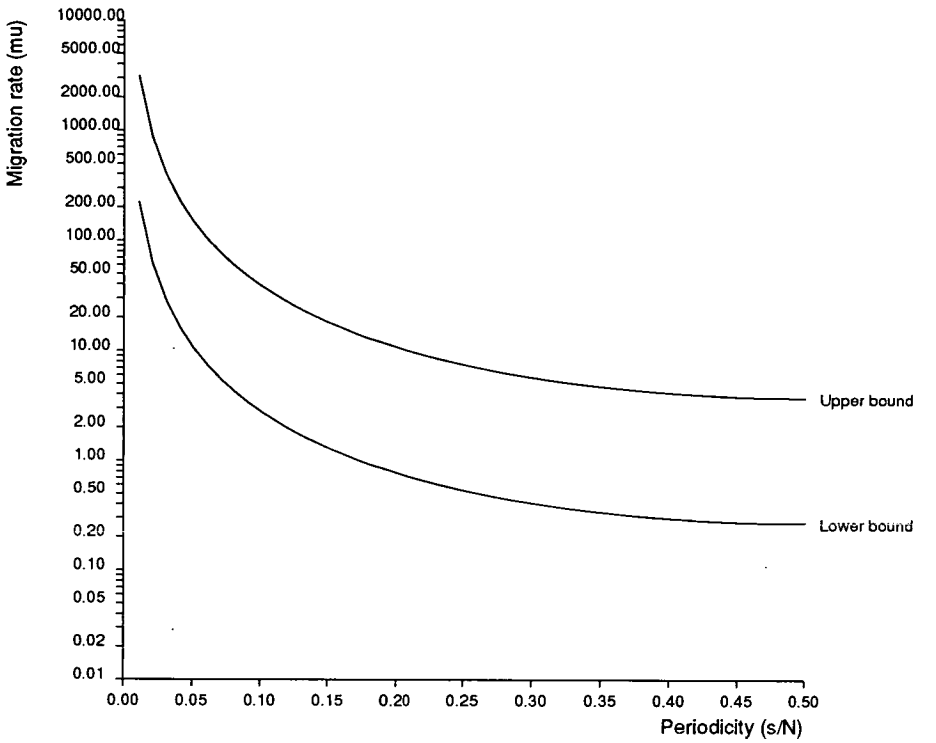

Figure 2.13: Oscillatory domain periodicity limits

Graphical representation of the upper and lower bands of the oscillatory region within (μ, λ')-space for the linearised, one-dimensional two-reactant Turing system. This represents the most general kilter divisions, with the whole of the oscillatory region contained within realisable parameter space. This example uses set parameter values of $a = m = 2, d = -2$ and $bc = -3$.

- If $a < d$ then
  - lower bound, $U_1 = m(a - d - 2\sqrt{-bc})/(m - 1)$ is always negative, and
  - upper bound, $U_2 = m(a-d+2\sqrt{-bc})/(m-1)$ is positive if $2\sqrt{-bc} > |a-d|$.
- If $a > d$ then
  - lower bound, $U_1 = m(a-d-2\sqrt{-bc})/(m-1)$ is positive if $2\sqrt{-bc} < a-d$, and
  - upper bound, $U_2 = m(a - d + 2\sqrt{-bc})/(m - 1)$ is always positive.

Therefore the lower bound on the oscillatory region will always equate to $U = 0$ if we choose $a \leq d$, and we will always have a real upper bound (i.e. the co-existence of both kilters within possible (μ, λ')-space) if we choose $a \geq d$. We have the possibility of no kilter divisions, and hence all possible solutions are stationary, if $a < d$ and $2\sqrt{-bc} \leq |a - d|$. Whilst with $a > d$ and $a - d > 2\sqrt{-bc}$ we will have both upper

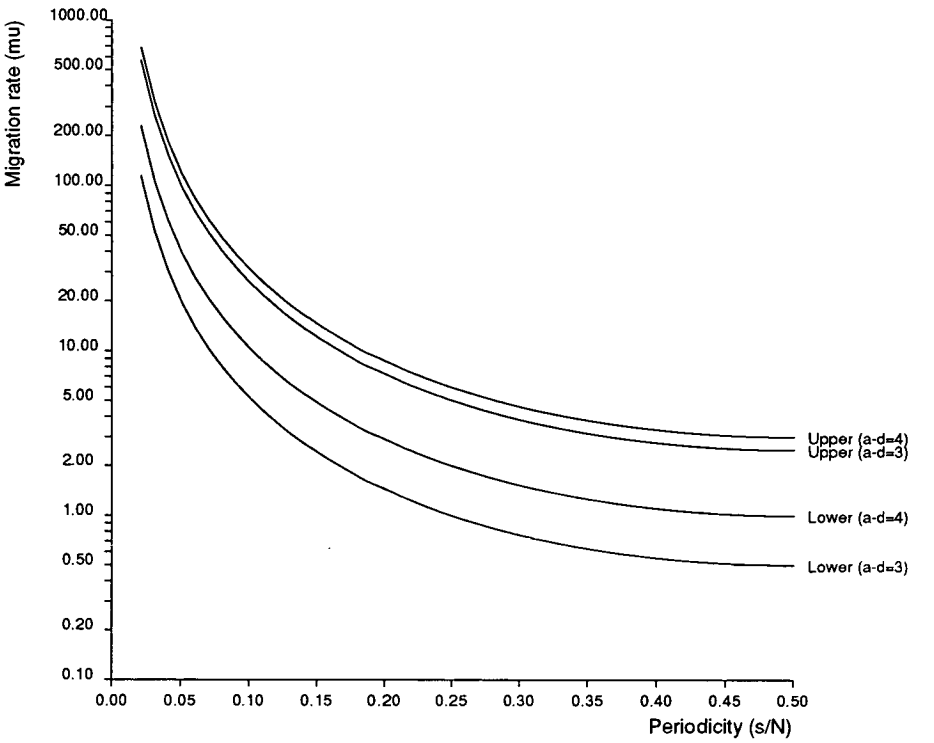

Figure 2.14: Variation of the oscillatory domain boundaries

Graphical representation of the upper and lower bands of the oscillatory region within (μ, λ')-space for the linearised, one-dimensional two-reactant Turing system for varying values of $|a - d|$. This example uses set parameter values of $a = m = 2, d = -2$ and $bc = -3$, and shows how the bounds rise with increasing $|a - d|$.

and lower bounds within (μ, λ')-space. This final alternative is the most interesting and gives us the possibility of a band of oscillatory parameters entirely contained within physically realisable parameter space for stationary activity. An example of such a scenario is given in Figure 2.13.

Given such a general scenario we can also investigate how the nature of this oscillatory band is affected by varying the chosen values of a, b, c, d and m. We need not consider each of these five alternatives individually, since from (2.16) the bounds are dependent upon only $|bc|, m$ and $|a - d|$. Dealing with the latter dependency first, it is straightforward to see that as the difference between self-interaction coefficients, $|a - d|$, increases, the *width* of the oscillatory band remains constant; whereas its position with respect to the migration rate increases in linear proportion to μ (this can be seen in Figure 2.14).

Variation of the migration rate ratio, m, and the product of cross-interaction coefficients $|bc|$, not only affects the position of the oscillatory band, but also changes its width. It was noted earlier that increasing $|bc|$ would widen the oscillatory region, this expansion

being in proportion to $\sqrt{-bc}$ around a central μ-value (all other variables being held fixed). Alternatively, increasing m will decrease both the width and position of the band relative to μ in proportion to $m/(m-1)$. Figure 2.15 shows both these effects on a single graph. Each of the three sets of five curves represents the system for a particular value of migration ratio, showing the variation as $m \to 1$; within each set we see the increase in the size of the oscillatory kilter region as $|bc|$ grows and we move away from a central single kilter curve. Equation 2.16 shows that the band width in U, and hence the amount of available parameter space responsible for oscillatory motion, is

$$\frac{4m}{m-1}\sqrt{-bc} \quad \text{centred around} \quad U = \frac{m}{m-1}(a-d) \ .$$

An Extended Criticality Specification

Now we have defined the general bounds of oscillatory and stationary activity, we can also study the criticality surface within each area. Following the same principles used earlier for our specific set of parameters, let us define the surfaces within each kilter domain, and then unify these for the whole of the parameter space.

Oscillatory kilter: From (2.11) we obtain critical behaviour when

$$U = 4\mu \sin^2(\pi\lambda') = m(a+d)/(m+1) \ . \tag{2.17}$$

This value is independent of bc, and produces a surface very similar to that shown in Figure 2.7 if we regard the vertical axis to represent the value $a+d$. We can thus determine those a and d values that will provide critical behaviour for each point in (μ, λ')-space, although we must remember that only a certain band of this space will actually represent oscillatory kilter. As we vary m within this scenario the criticality surface will be scaled in proportion to $(m+1)/m$.

Stationary kilter: Here the criticality can also be deduced from (2.11), and is slightly more complex. Critical behaviour will occur when system parameters are chosen such that

$$\frac{a+d}{2} - U\frac{m+1}{2m} + \frac{1}{2}\sqrt{4bc + \left(\frac{U}{m}(m-1) + (d-a)\right)^2} = 0 \ .$$

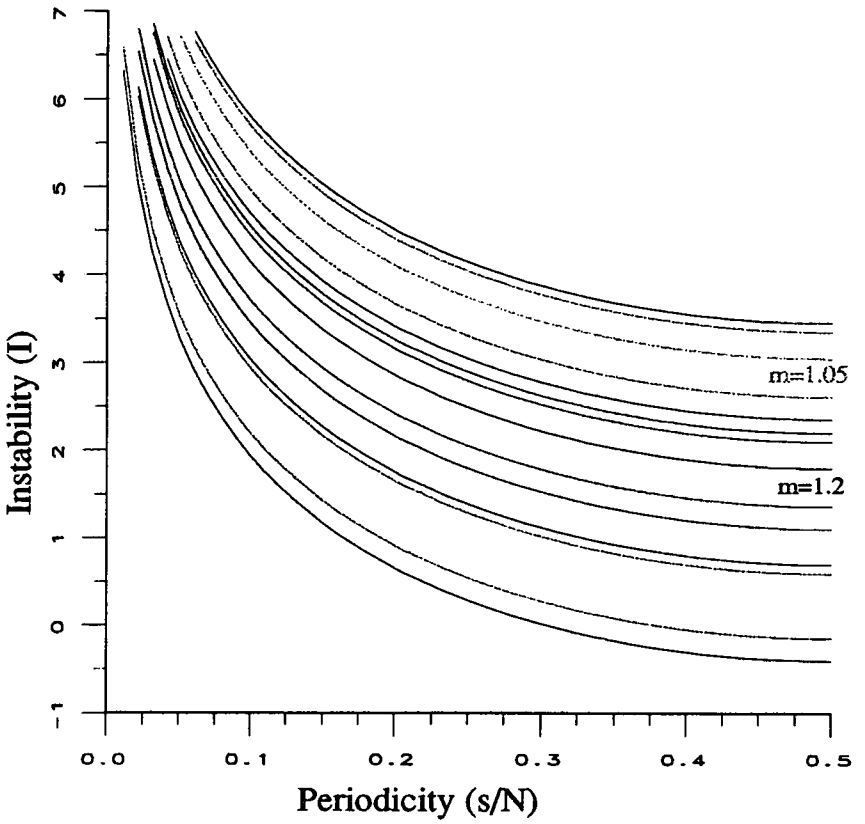

Figure 2.15: Possible variations in oscillatory domain size

A graphical representation of the variation in position and width of the upper and lower bands of the oscillatory region within (μ, λ')-space for the linearised, one-dimensional two-reactant Turing system, given a range of m and $|bc|$ values. This example uses set parameter values of $a = 2$ and $d = -2$, and shows how the bands widen with increasing $|bc|$, i.e. colour shading increases. For each m value we have five curves. The central (lightest shaded) line represents $bc = 0$, the two extreme lines (darkest shading) represent $bc = -1$, and the two lines between these correspond to $bc = -0.5$. In addition, we see that as $m \rightarrow 1$ the oscillatory band lies at increasing values of migration rate μ.

This expression can be simplified substantially to give

$$U^2 - U(md + a) + m(ad + bc) = 0 \,, \tag{2.18}$$

which provides the roots

$$U_{1,2} = (md + a)/2 \pm 1/2\sqrt{(md - a)^2 - 4mbc} \,,$$

that produce critical behaviour.

In order to define a complete unified criticality specification, let us examine Equation (2.18) and reorganise the terms to give us an expression for one interaction coefficient with respect to all other system parameters. If we choose a, for example, we obtain

$$a = U - bc/(d - U/m)$$

at criticality, as compared to

$$a = U(m + 1)/m - d$$

taken from (2.17) for the oscillatory kilter. Figure 2.16 shows the criticality surface for a given certain choices of parameter as detailed. We see that this surface is very similar to that previously described in Figure 2.12, except that we now have both kilter divisions within the physically realistic parameter space.

Equations (2.16) and (2.18) fully describe the criticality and kilter for the most general parameter space of a linearised one-dimensional spatial reaction system. We can therefore predict expected behaviour directly from any interaction and migration/diffusion coefficients, and in addition, can identify the changes in behaviour produced by any parameter variations. This is of particular importance in assessing the criticality of systems close to stability, if certain parameters are subject to some uncertainty. We will return to this subject when studying the stochastic system in Chapter 4.

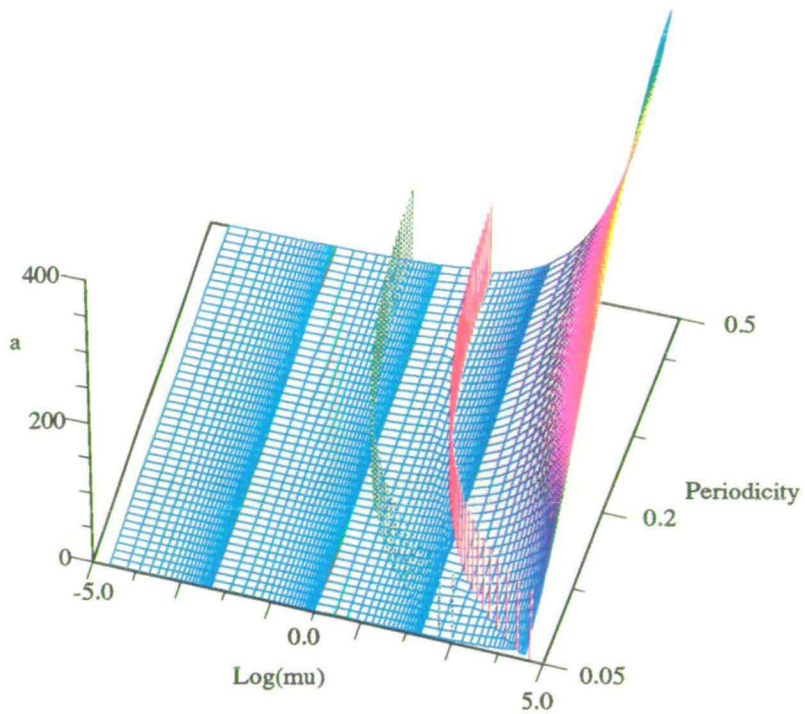

Figure 2.16: Parameter values to achieve criticality

The unified criticality surface for a in the linearised, one-dimensional two-reactant Turing system with $bc = -3, m = 4$ and $d = -2$. The graph shows the necessary value of a to achieve critical behaviour. The two impulse plots highlight the upper and lower bounds of the region of oscillatory kilter. This graphs shows that the unified surface is continuous at these divisions.

2.4 Systems with Non-Linear Interactions

As stated at the start of this chapter, any scientist wishing to study real-world systems must, in general, consider non-linear functions of interaction between system reactants. In an excellent review paper Mollison [1977] describes the importance of studying full non-linear systems, a view expressed lucidly by May [1976] one year earlier. It is interesting to note that Mollison also stresses the inadequacy of diffusion based models, and points out that a deterministic model can only be approximately equivalent to a stochastic system in the linear case. Indeed, the importance of stochastic systems was realised in some of the earliest work in the ecological field; Fisher [1937] dealt with the deterministic case in the knowledge that this was an approximate substitute for stochastic phenomena. In more recent publications we have seen specific analytic results that show a much closer relationship between the non-linear deterministic and the linear stochastic scenarios (McKean [1975]), and this motivates us to attempt some form of analysis of the non-linear system.

However, analysis of non-linear systems is still in its infancy. In the biological field there have been only a small number of notable successes. The first of these was the saddle-point approximation technique of Daniels [1954], which he later applied specifically to ecological modelling [1977]. Much more recently Renshaw [1994] has applied truncated Fourier transforms to approximate non-linear Turing ring systems. His observations of non-linear wave patterns in reactant populations complement our results detailed later in Chapter 3. Renshaw [1977] has also worked in a third area of non-linear system analysis, following some early work of Mollison in epidemiology [1972a]. In this scenario, knowledge of the existence of a travelling wave structure within the expected solution allows the development of particular solutions to the non-linear system.

From our realisation studies of spatial reaction systems, we know that in the transient phase following a perturbation, the effect on reactant populations travels through the system in a wave-like manner (see Section 3.4). In addition, Turing's original work suggests that for scenarios with three reactants, given appropriate parameter choices, the linearised system produces travelling waves of reactant concentrations. We therefore have a strong motivation to investigate a travelling waveform approach for the solution of our general spatial reaction systems with non-linear interactions.

2.4.1 A Travelling Waveform Solution

In Mollison's ground-breaking work [1972a] a technique is introduced to obtain approximate travelling waveform shapes and velocities for simple spatial epidemic processes using first-order techniques. These results are then confirmed by Daniels [1975] using saddle-point approximations; later he strengthens the case for non-linear analysis as he cast doubts on the validity of general linearised deterministic results [1977]. At about the same time Renshaw [1977] produces exact expressions for wave propagation velocity and shape for birth-death-migration models in a one-dimensional stepping-stone environment. Renshaw [1981] then turns his attention to one-dimensional spatial epidemics with non-linear infection functions, and develops a new technique based on Laplacian power series expansions to determine the velocity of the wave of infection. He then applies this same approach to spatial predator-prey processes to describe the rate of spread of predators [1982].

We therefore see a host of literature (see Mollison [1977] for an early review) concerning the mathematical study of the velocity of population movement. This is accompanied by much application-based work, from the spread of oak trees by Skellam [1951] to the dispersion of innovations within spatial human geography models by Cliff & Ord [1975]. In general this past work has been concerned with the movement of a single type of entity within a spatial domain, and an attempt to define a *shape* and velocity (v) for the population density waves. It usually proves possible to define such a shape only when the velocity of propagation is above some minimum threshold v_0, and there then exists a set of possible waveforms, one for each possible velocity value v (see Mollison [1972b]).

In order to observe travelling waves in our spatial reaction scenario we need to advance current techniques to consider at least two reactants. Our analyses attempt to approach both the transient and persistent states of our spatial reaction systems, and the simulation work we have performed (detailed in Chapter 3) has been invaluable in allowing us to highlight areas of potential interest. In particular, graphical realisations of our one-dimensional systems (akin to those shown in later Figure 3.23) provide the motivation to look for transient travelling wave structures in the two-reactant systems, where previous analysis had pointed to more complex scenarios (e.g. at least three reactants) being necessary to produce such behaviour.

2.4.2 Systems with Two Reactants

In order to enable the study of non-linear systems, we can extend our consideration of linearised spatial reaction systems to include the following general, potentially unstable, local non-linear interaction functions as used by Renshaw [1991].

$$\mathcal{F}(X,Y) = X(r_1 + a_1X + b_1Y) \quad \text{and} \quad \mathcal{G}(X,Y) = Y(r_2 + a_2X + b_2Y) \ . \tag{2.19}$$

Concentrating on the dynamics of the general system at each individual site, thus ignoring migration between cells, we can denote the populations of the two species in a cell to be $X(t)$ and $Y(t)$. We know from (2.19) that the deterministic equations for the system are

$$dX(t)/dt = X(t)[r_1 + a_1X(t) + b_1Y(t)] \quad \text{and} \tag{2.20}$$

$$dY(t)/dt = Y(t)[r_2 + a_2X(t) + b_2Y(t)] \ .$$

For a positive equilibrium solution (X^*, Y^*) to exist we must clearly have

$$X^*[r_1 + a_1X^* + b_1Y^*] = 0 = Y^*[r_2 + a_2X^* + b_2Y^*] \ , \tag{2.21}$$

from which we can deduce

$$X^* = \frac{b_1r_2 - b_2r_1}{a_1b_2 - a_2b_1} \quad \text{and} \quad Y^* = \frac{a_1r_2 - a_2r_1}{a_2b_1 - a_1b_2} \ . \tag{2.22}$$

If we consider equations (2.1) and (2.19) with interaction functions in their most general non-linear form, we can attempt to expand the techniques of Renshaw [1982] for this system. Let us therefore assume that we can describe the reactant populations in terms of a simple Laplace transform-type power series expansion. Assuming the waveform is travelling through the spatial domain at constant velocity, a tangible analytic solution requires a change in the frame of reference for the system to one that moves at constant velocity (v) with the wave front. We can specify this change of reference frame by denoting $s = r - vt$. Providing we ensure that t is small relative to N (so that the ring system approximates to an infinite line, for a local initial perturbation from equilibrium) we can denote

$$m(s) = X_{s+vt}(t) \quad \text{and} \quad n(s) = Y_{s+vt}(t) \ .$$

These expressions can now be substituted into (2.1) and (2.19) to give the following representation of the non-linear system

$$-vm'(s) = m(s)[r_1 + a_1 m(s) + b_1 n(s)] + \mu[m(s-1) - 2m(s) + m(s+1)] \quad (2.23)$$

$$-vn'(s) = n(s)[r_2 + a_2 m(s) + b_2 n(s)] + \nu[n(s-1) - 2n(s) + n(s+1)] \ . \quad (2.24)$$

Here $(')$ represents differentiation with respect to s, with $d/dt = -v\ d/ds$.

Let us introduce the Laplace transform-type power series expansions

$$m(s) = \sum_{p=0}^{\infty} \alpha_p e^{-p\theta s} \quad \text{and} \quad n(s) = \sum_{p=0}^{\infty} \beta_p e^{-p\theta s}, \quad (2.25)$$

where θ is real and positive and the coefficients α_p and β_p are unknown constants. Substituting (2.25) into (2.23) and (2.24) gives

$$v\sum_{p=0}^{\infty} \alpha_p p\theta e^{-p\theta s} = (r_1 - 2\mu)\sum_{p=0}^{\infty} \alpha_p e^{-p\theta s} + a_1 \sum_{p=0}^{\infty} \alpha_p e^{-p\theta s} \sum_{p=0}^{\infty} \alpha_p e^{-p\theta s} +$$
$$b_1 \sum_{p=0}^{\infty} \alpha_p e^{-p\theta s} \sum_{p=0}^{\infty} \beta_p e^{-p\theta s} + \mu \sum_{p=0}^{\infty} \alpha_p e^{-p\theta(s-1)} + \mu \sum_{p=0}^{\infty} \alpha_p e^{-p\theta(s+1)} \quad (2.26)$$

and

$$v\sum_{p=0}^{\infty} \beta_p p\theta e^{-p\theta s} = (r_2 - 2\nu)\sum_{p=0}^{\infty} \beta_p e^{-p\theta s} + a_2 \sum_{p=0}^{\infty} \alpha_p e^{-p\theta s} \sum_{p=0}^{\infty} \beta_p e^{-p\theta s} +$$
$$b_2 \sum_{p=0}^{\infty} \beta_p e^{-p\theta s} \sum_{p=0}^{\infty} \beta_p e^{-p\theta s} + \nu \sum_{p=0}^{\infty} \beta_p e^{-p\theta(s-1)} + \nu \sum_{p=0}^{\infty} \beta_p e^{-p\theta(s+1)} \ . \quad (2.27)$$

Expressions (2.26) and (2.27) can now be used to obtain values for the coefficients $\{\alpha_p\}$ and $\{\beta_p\}$ by the extraction of the coefficients of the $e^{-p\theta s}$ terms for successive values of p. Starting with $p = 0$ we obtain

$$r_1\alpha_0 + a_1\alpha_0^2 + b_1\alpha_0\beta_0 = 0 \quad \text{and} \quad r_2\beta_0 + a_2\alpha_0\beta_0 + b_2\beta_0^2 = 0 \ . \quad (2.28)$$

These relations can be valid for two distinct cases: we have termed these *o-waves* and *e-waves* as they represent waveforms travelling through an empty ring and a ring at equilibrium respectively.

o-waves: We have a simple special case of (2.28) when $\alpha_0 = \beta_0 = 0$. In Ren-

shaw [1982] this corresponds to the case of zero populations in all cells ahead of the wavefront. Simulation details given later (Section 3.4.3) show that in sub-critical scenarios both reactants can spread out from a single cell (given a sufficiently large initial perturbation) to populate the whole system.

Extraction of the coefficients of $e^{-p\theta s}$ when $p = 1$ from (2.26) and (2.27) produces

$$c\alpha_1\theta = (r_1 - 2\mu)\alpha_1 + \mu(\alpha_1 e^{\theta} + \alpha_1 e^{-\theta}) \tag{2.29}$$

and

$$c\beta_1\theta = (r_2 - 2\nu)\beta_1 + \nu(\beta_1 e^{\theta} + \beta_1 e^{-\theta}). \tag{2.30}$$

However, when we attempt to solve (2.29) and (2.30) for v and θ we find that we obtain the straightforward equality

$$v\theta = \frac{\mu r_2 - \nu r_1}{\mu - \nu},$$

which upon differentiation with respect to θ produces

$$v + \theta\frac{dc}{d\theta} = 0.$$

So if we wish to obtain a constant wave velocity (i.e $dv/d\theta = 0$), we find that this is impossible for any real value of v.

e-waves: In this more general case we consider that either, or both, $\alpha_0 \neq 0$ and $\beta_0 \neq 0$. This allows some simplification of the relations (2.28), namely

$$\alpha_0 = \frac{b_1 r_2 - b_2 r_1}{a_1 b_2 - a_2 b_1} \quad \text{and} \quad \beta_0 = \frac{a_2 r_1 - a_1 r_2}{a_1 b_2 - a_2 b_1}. \tag{2.31}$$

It should be noted that these expressions are equivalent to (2.22) and therefore correspond to the equilibrium populations of each reactant in the linearised version of the system. Simulations show that we observe travelling wave motion for such equilibrium-start scenarios during the transient phase.

With defined values for both α_0 and β_0, and we can look again at equations (2.26)

and (2.27) and extract coefficients of $e^{-p\theta s}$ when $p = 1$, namely

$$v\alpha_1\theta = (r_1 - 2\mu)\alpha_1 + 2a_1\alpha_0\alpha_1 + b_1(\alpha_0\beta_1 + \alpha_1\beta_0) + \mu(\alpha_1 e^{\theta} + \alpha_1 e^{-\theta}) \quad (2.32)$$

$$v\beta_1\theta = (r_2 - 2\nu)\beta_1 + a_2(\alpha_0\beta_1 + \alpha_1\beta_0) + 2b_2\beta_0\beta_1 + \nu(\beta_1 e^{\theta} + \beta_1 e^{-\theta}). \quad (2.33)$$

The above relations can be used to generate successive values of the power series coefficients $\{\alpha_p\}$ and $\{\beta_p\}$, given known values for v and θ. However, when we attempt to manipulate equations (2.32) and (2.33) to give an expression of the waveform velocity v in terms of θ, we again obtain expressions with no solutions when $dv/d\theta = 0$ for real v.

We therefore see that the constant waveform approximation *cannot* account for the transient wave activity in the two-reactant scenario.

2.4.3 Systems with Three Reactants

Since Turing claims that the three-reactant scenario produces travelling reactant waves in the linear approximation, let us consider the full non-linear three-reactant case under the constant waveform approximation. Unfortunately we find that although this approach does yield an analytic result, numerical solution using parameter values known to produce travelling wave solutions shows that the analytic wave velocity lies in the complex domain. However, the process of reaching this result is informative. Therefore, if we again consider equation (2.1), we must first form a specification for the non-linear interaction functions $\mathcal{F}^c$ for the three-reactant case. Let us follow the same pattern as in Section 2.4.2 and use the following functions, with W_r representing the population/concentration of the third reactant type in cell r:

$$\mathcal{F}(X_r, Y_r, W_r) = X_r(r_1 + a_1X_r + b_1Y_r + c_1W_r), \quad (2.34)$$

$$\mathcal{G}(X_r, Y_r, W_r) = Y_r(r_2 + a_2X_r + b_2Y_r + c_2W_r), \quad \text{and} \quad (2.35)$$

$$\mathcal{H}(X_r, Y_r, W_r) = W_r(r_3 + a_3X_r + b_3Y_r + c_3W_r)\ . \quad (2.36)$$

Inserting these functions into (2.1) for a one-dimensional system, we obtain equations that are very difficult to handle analytically. Using Turing's linearisation approach we

can study the system as a simple extension of the two-reactant case in Section 2.2. This approach gives us equations for perturbations from an equilibrium state (X^*, Y^*, W^*) as:

$$dx_r/dt = ax_r + by_r + cw_r + \mu(x_{r+1} - 2x_r + x_{r-1})$$

$$dy_r/dt = dx_r + ey_r + fw_r + \nu(y_{r+1} - 2y_r + y_{r-1})$$

$$dw_r/dt = gx_r + hy_r + jw_r + \vartheta(w_{r+1} - 2w_r + w_{r-1}) \ .$$

Introducing the Fourier transforms

$$x_r = \sum_{s=0}^{N-1} \zeta_s e^{2\pi irs/N}, \quad y_r = \sum_{s=0}^{N-1} \eta_s e^{2\pi irs/N}, \quad \text{and} \quad w_r = \sum_{s=0}^{N-1} \xi_s e^{2\pi irs/N},$$

we obtain

$$d\zeta_s/dt = [a - 4\mu \sin^2(\pi s/N)]\zeta_s + b\eta_s + c\xi_s,$$

$$d\eta_s/dt = d\zeta_s + [e - 4\nu \sin^2(\pi s/N)]\eta_s + f\xi_s, \quad \text{and}$$

$$d\xi_s/dt = g\zeta_s + h\eta_s + [j - 4\vartheta \sin^2(\pi s/N)]\xi_s$$

using methods extended from those in Section 2.2.1.

Let us again look for a non-linear solution for the reactant populations in terms of a simple Laplace transform-type power series expansion, Assuming that this waveform is moving with a constant velocity v, and again specifying a change of reference frame by denoting $s = r - vt$, we can denote

$$m(s) = X_{s+vt}(t) \ , \quad n(s) = Y_{s+vt}(t) \quad \text{and} \quad l(s) = W_{s+vt}(t) \ .$$

Substituting these expressions into (2.1), together with the non-linear functions above, yields

$$\begin{aligned}
-vm'(s) &= m(s)[r_1 + a_1 m(s) + b_1 n(s) + c_1 l(s)] + \mu[m(s-1) - 2m(s) + m(s+1)] \\
-vn'(s) &= n(s)[r_2 + a_2 m(s) + b_2 n(s) + c_2 l(s)] + \nu[n(s-1) - 2n(s) + n(s+1)] \\
-vl'(s) &= l(s)[r_3 + a_3 m(s) + b_3 n(s) + c_3 l(s)] + \vartheta[l(s-1) - 2l(s) + l(s+1)] \ ,
\end{aligned} \tag{2.37}$$

where $(')$ again represents differentiation with respect to s, and $d/dt = -c\, d/ds$.

To solve Equations (2.37) we introduce the following Laplace transform-type power series expansions

$$m(s) = \sum_{p=0}^{\infty} \alpha_p e^{-p\theta s}\ ,\ n(s) = \sum_{p=0}^{\infty} \beta_p e^{-p\theta s} \quad \text{and} \quad l(s) = \sum_{p=0}^{\infty} \gamma_p e^{-p\theta s}, \tag{2.38}$$

where θ is real and positive and α_p, β_p and γ_p are unknown coefficients. Substituting (2.38) into (2.37) gives

$$\begin{aligned} v\sum_{p=0}^{\infty} \alpha_p p\theta e^{-p\theta s} &= (r_1 - 2\mu)\sum_{p=0}^{\infty} \alpha_p e^{-p\theta s} + a_1 \sum_{p=0}^{\infty} \alpha_p e^{-p\theta s} \sum_{p=0}^{\infty} \alpha_p e^{-p\theta s} \\ &+ b_1 \sum_{p=0}^{\infty} \alpha_p e^{-p\theta s} \sum_{p=0}^{\infty} \beta_p e^{-p\theta s} + c_1 \sum_{p=0}^{\infty} \alpha_p e^{-p\theta s} \sum_{p=0}^{\infty} \gamma_p e^{-p\theta s} \\ &+ \mu \sum_{p=0}^{\infty} \alpha_p e^{-p\theta(s-1)} + \mu \sum_{p=0}^{\infty} \alpha_p e^{-p\theta(s+1)}, \end{aligned} \tag{2.39}$$

$$\begin{aligned} v\sum_{p=0}^{\infty} \beta_p p\theta e^{-p\theta s} &= (r_2 - 2\nu)\sum_{p=0}^{\infty} \beta_p e^{-p\theta s} + a_2 \sum_{p=0}^{\infty} \alpha_p e^{-p\theta s} \sum_{p=0}^{\infty} \beta_p e^{-p\theta s} \\ &+ b_2 \sum_{p=0}^{\infty} \beta_p e^{-p\theta s} \sum_{p=0}^{\infty} \beta_p e^{-p\theta s} + c_2 \sum_{p=0}^{\infty} \gamma_p e^{-p\theta s} \sum_{p=0}^{\infty} \beta_p e^{-p\theta s} \\ &+ \nu \sum_{p=0}^{\infty} \beta_p e^{-p\theta(s-1)} + \nu \sum_{p=0}^{\infty} \beta_p e^{-p\theta(s+1)}, \end{aligned} \tag{2.40}$$

$$\begin{aligned} v\sum_{p=0}^{\infty} \gamma_p p\theta e^{-p\theta s} &= (r_3 - 2\vartheta)\sum_{p=0}^{\infty} \gamma_p e^{-p\theta s} + a_3 \sum_{p=0}^{\infty} \alpha_p e^{-p\theta s} \sum_{p=0}^{\infty} \gamma_p e^{-p\theta s} \\ &+ b_3 \sum_{p=0}^{\infty} \beta_p e^{-p\theta s} \sum_{p=0}^{\infty} \gamma_p e^{-p\theta s} + c_3 \sum_{p=0}^{\infty} \gamma_p e^{-p\theta s} \sum_{p=0}^{\infty} \gamma_p e^{-p\theta s} \\ &+ \vartheta \sum_{p=0}^{\infty} \gamma_p e^{-p\theta(s-1)} + \vartheta \sum_{p=0}^{\infty} \gamma_p e^{-p\theta(s+1)}. \end{aligned} \tag{2.41}$$

Expressions (2.39), (2.40) and (2.41) can now be used to obtain values for the coefficients $\{\alpha_p\}, \{\beta_p\}$ and $\{\gamma_p\}$ by extracting coefficients of $e^{-p\theta s}$ for successive values of p. With $p = 0$ we obtain

$$r_1\alpha_0 + a_1\alpha_0^2 + b_1\alpha_0\beta_0 + c_1\alpha_0\gamma_0 = 0, \tag{2.42}$$

$$r_2\beta_0 + a_2\beta_0\alpha_0 + b_2\beta_0^2 + c_2\beta_0\gamma_0 = 0, \quad \text{and} \tag{2.43}$$

$$r_3\gamma_0 + a_3\gamma_0\alpha_0 + b_3\gamma_0\beta_0 + c_1\gamma_0^2 = 0\ . \tag{2.44}$$

These three relations produce unique values for the leading terms in the power series expansions $(\alpha_0, \beta_0, \gamma_0)$, provided that we ensure that all remain non-zero. This assumption allows us to simplify the above equations to give

$$\begin{aligned} r_1 + a_1\alpha_0 + b_1\beta_0 + c_1\gamma_0 &= 0, \\ r_2 + a_2\alpha_0 + b_2\beta_0 + c_2\gamma_0 &= 0, \quad \text{and} \\ r_3 + a_3\alpha_0 + b_3\beta_0 + c_1\gamma_0 &= 0 \ . \end{aligned} \tag{2.45}$$

These relations can then be manipulated to give exact determinant-like expressions for the leading terms in the power series expansions in terms of the interaction coefficients, e.g.

$$\alpha_0 = \frac{(r_1c_2 - r_2c_1)(b_3c_2 - b_2c_3) - (r_2c_3 - r_3c_2)(b_2c_1 - b_1c_2)}{(c_1a_2 - c_2a_1)(b_3c_2 - b_2c_3) - (c_2a_3 - c_3a_2)(b_2c_1 - b_1c_2)}, \tag{2.46}$$

and similarly for β_0 and γ_0. By considering our three-reactant extension to the linearisation approximations given in (2.21) in Section 2.4.2, we can see that equations (2.45) are identical to this extension if we regard $\alpha_0 = X^*, \beta_0 = Y^*$ and $\gamma_0 = W^*$. Therefore these leading terms can be taken to equate to the non-linear global equilibrium populations, and hence it is valid to ignore scenarios where $\alpha_0, \beta_0, \gamma_0 = 0$.

Using these defined values for α_0, β_0 and γ_0, we can examine Equations (2.39), (2.40) and (2.41) to extract coefficients of $e^{-p\theta s}$ when $p = 1$, namely

$$\begin{aligned} v\alpha_1\theta &= (r_1 - 2\mu)\alpha_1 + a_1(\alpha_0\alpha_1 + \alpha_1\alpha_0) + b_1(\alpha_0\beta_1 + \alpha_1\beta_0) \\ &+ c_1(\alpha_0\gamma_1 + \alpha_1\gamma_0) + \mu(\alpha_1 e^{\theta} + \alpha_1 e^{-\theta}) \ , \end{aligned} \tag{2.47}$$

$$\begin{aligned} v\beta_1\theta &= (r_2 - 2\nu)\beta_1 + a_2(\alpha_0\beta_1 + \alpha_1\beta_0) + b_2(\beta_0\beta_1 + \beta_1\beta_0) \\ &+ c_2(\beta_0\gamma_1 + \beta_1\gamma_0) + \nu(\beta_1 e^{\theta} + \beta_1 e^{-\theta}), \quad \text{and} \end{aligned} \tag{2.48}$$

$$\begin{aligned} v\gamma_1\theta &= (r_3 - 2\vartheta)\gamma_1 + a_3(\alpha_0\gamma_1 + \alpha_1\gamma_0) + b_3(\beta_0\gamma_1 + \beta_1\gamma_0) \\ &+ c_3(\gamma_0\gamma_1 + \gamma_1\gamma_0) + \vartheta(\gamma_1 e^{\theta} + \gamma_1 e^{-\theta}) \ . \end{aligned} \tag{2.49}$$

Similarly with $p > 1$ we have the general relations of the form

$$\begin{aligned} vp\alpha_p\theta &= (r_1 - 2\mu)\alpha_p + a_1(\alpha_0\alpha_p + \alpha_1\alpha_{p-1} + \cdots + \alpha_{p-1}\alpha_1 + \alpha_p) \\ &+ b_1(\alpha_0\beta_p + \alpha_1\beta_{p-1} + \cdots + \alpha_{p-1}\beta_1 + \alpha_p\beta_0) \end{aligned} \tag{2.50}$$

$$+ \quad c_1(\alpha_0\gamma_p + \alpha_1\gamma_{p-1} + \cdots + \alpha_{p-1}\gamma_1 + \alpha_p\gamma_0)$$
$$+ \quad \mu(\alpha_p e^{p\theta} + \alpha_p e^{-p\theta}) \ .$$

Equations (2.47), (2.48) and (2.49) can be simplified using the expressions in (2.45) to give

$$\alpha_1(v\theta - a_1\alpha_0 - \mu(e^{\theta} - 2 + e^{-\theta})) = b_1\alpha_0\beta_1 + c_1\alpha_0\gamma_1,$$

$$\beta_1(v\theta - b_2\beta_0 - \nu(e^{\theta} - 2 + e^{-\theta})) = a_2\beta_0\alpha_1 + c_2\beta_0\gamma_1,$$

$$\gamma_1(v\theta - c_3\gamma_0 - \vartheta(e^{\theta} - 2 + e^{-\theta})) = a_3\gamma_0\alpha_1 + b_3\gamma_0\beta_1 \ .$$

Given these three equations in the three unknowns α_1, β_1 and γ_1 we can manipulate the relations to produce expressions unique in each unknown. Thus for α_1 we obtain

$$A\alpha_1 = b_1\beta_0\alpha_1\frac{Ac_2 + a_2c_1\alpha_0}{b_1c_2\beta_0 + Bc_1} + c_1\gamma_0\alpha_1\frac{Ab_3 + a_3b_1\alpha_0}{Cb_1 + c_1b_3\gamma_0}, \tag{2.51}$$

where

$$\begin{aligned} A &= v\theta - a_1\alpha_0 - \mu(e^{\theta} - 2 + e^{-\theta}), \\ B &= v\theta - b_2\beta_0 - \nu(e^{\theta} - 2 + e^{-\theta}), \quad \text{and} \\ C &= v\theta - c_3\gamma_0 - \vartheta(e^{\theta} - 2 + e^{-\theta}) \ . \end{aligned}$$

Provided we ensure that α_1 is non-zero, (2.51) reverts into an expression where the only unknowns are v and θ, namely

$$\begin{aligned} v\theta - a_1\alpha_0 - \mu(\theta) &= \frac{b_1c_2\beta_0(v\theta - a_1\alpha_0 - \mu(\theta) + a_2c_1\alpha_0/c_2)}{c_1(v\theta - b_2\beta_0 - \nu(\theta) + b_1c_2\beta_0/c_1)} \\ &+ \frac{c_1b_3\gamma_0(v\theta - a_1\alpha_0 - \mu(\theta) + a_3b_1\alpha_0/b_3)}{b_1(v\theta - c_3\gamma_0 - \vartheta(\theta) + b_3c_1\gamma_0/b_1)} \ . \end{aligned} \tag{2.52}$$

This expression can be differentiated with respect to θ, whence on setting $dv/d\theta = 0$ we can look for values of v and θ that satisfy the system. If we investigate one particular three-reactant scenario in which Turing predicted the presence of travelling waves, we see that unfortunately the Laplace transform approximation fails again. Using Turing's linear parameter set (with $I = 0$) we obtain the following coefficients for the non-linear

system;

$$a_1 = -1/3,\ \ b_1 = 3/10,\ \ c_1 = -1/10,\ \ r_1 = 4/3,\ \ \mu = 2/3$$

$$a_2 = -1/5,\ \ b_2 = 7/30,\ \ c_2 = 0,\ \ r_2 = -1/3,\ \ \nu = 1/3$$

$$a_3 = 3/10,\ \ b_3 = -2/5,\ \ c_3 = 0,\ \ r_3 = 1,\ \ \vartheta = 0\ .$$

These allow us to reduce (2.52) to

$$3(v\theta)^2 + 14v\theta - 2v\theta(e^{\theta} + e^{-\theta}) + 90 = 0, \tag{2.53}$$

which differentiates with respect to θ to give

$$6v\theta = 2e^{\theta}(1+\theta) + 2e^{-\theta}(1-\theta) - 14 \tag{2.54}$$

after setting $dv/d\theta = 0$ and assuming $v \neq 0$.

Numerical solution of (2.53) and (2.54) reveal that there is just one solution, but this lies in the complex domain ($\theta = 1.8 - 1.15i, v = 0.66 - 2.98i$). We therefore have no real wave velocity for which this approximation to the non-linear system is valid. This is an unfortunate result, since our computer realisations clearly show the existence of travelling waves during the transient phase of system development. However, if we consider long-term behaviour, the results detailed later in Section 3.5 highlight an important difference between Turing's predictions and actual realisation results. What we observe for the scenario selected above is the development of standing waves of reactant populations, and clearly not the travelling waves predicted. We therefore see that we have a situation in which current non-linear techniques cannot adequately describe system behaviour, and past linearised analysis has also failed to predict the result of either linearised or full non-linear computer realisations.

It is clear that further study is required in this domain, and investigation of other non-linear techniques may prove useful. In this work we feel that we have progressed our understanding of the general behaviour of the linearised scenario, in particular in terms of system criticality. The knowledge gained has proved vital for the next stage of study — deterministic realisations of linear and non-linear spatial reaction systems.

3

Numerical Realisation of Deterministic Spatial Reaction Systems

Mathematics, viewed rightly, possesses not only truth, but supreme beauty — a beauty cold and austere, like that of sculpture.

– Bertrand Russell

3.1 Introduction to Computer Realisation

Since their inception in the 1940s, computers have provided mathematicians and other scientists with a method of accelerating arithmetic calculation. It is interesting to connect the inventive mathematical genius of Alan Turing (so much discussed in previous chapters) with his pioneering work in the field of Computer Science and Computability [1936], for which he is perhaps more famous. The broad functionality offered by todays desk-top computers or workstations can often make it difficult to believe that these machines are still only performing arithmetic or logical operations — thereby assisting us with our mathematics. Turing had great interest in the development of computing because of his insight into how such machines could help with his mathematical investigations (see Hodges [1983]). The continued efforts of computer scientists since Turing's time have meant that computers become ever more usable and powerful.

The development of high-resolution workstation displays, coupled with straightforward programming languages and graphics libraries, allows modern-day mathematicians to visualise the results of calculations in a matter of seconds. This not only allows study of more systems in a given time-scale, but it also allows investigators to concentrate immediately on areas that appear to produce interesting results. This chapter details our investigations into the behaviour of various deterministic spatial reaction systems. The work has its foundation in the analytic studies of Chapter 2.

The basis of computer realisations of deterministic systems lies in the numerical solution of the equations that describe system development. Techniques for accurate numerical solutions are well established (see Rice [1983]), particularly in the field of non-linear differential equations (see Ortega [1970]). The main approach of all such techniques is to start from some given initial conditions, and then make some first-order approximation (based on given differential equations) of the state of the system after some small time increment (δt). The accuracy of this approximation increases as we consider smaller values of δt, and can be further improved by using higher-order techniques (e.g. Runge-Kutta as covered by Gear [1971]) to improve our first-order estimates. This pursuit of accuracy can be very computationally expensive, potentially adding orders of magnitude to the time taken to reach a certain point in a realisation. However, we must always ensure that our solutions are accurate enough to give a true representation of system development. Any inaccuracy in the predicted state of a system after a given time-step

can be compounded at the next iteration, as predicted values are used as the base for the next set of approximations. Errors that creep into the solution can thus grow very large, in the worst case they can dominate the true solution entirely. This is especially true when dealing with non-linear equations. For this reason we are always vigilant for approximation errors, typically caused by using too large a value of δt with a low-order solution technique, as well as for rounding errors that are inherent with any digital computer solution.

Various implementations of deterministic spatial reaction systems have been produced in the course of this work, and used to provide the results detailed in this and later chapters. These programs run on a number of parallel and serial machines, and were designed to provide a flexible user interface, and to offer a graphical display of realisation results. Recent developments in computer network technology have provided users with the ability to generate and visualise results on office workstations, making connections to supercomputers when required. This provides a powerful and flexible working environment. The majority of the one-dimensional results detailed later in Section 3.2 were, for example, performed on a serial Sun workstation, or sometimes on a Silicon Graphics two-processor graphics workstation, since the models are small enough to allow such machines to calculate accurate solutions within very short time periods (i.e. a few minutes). In contrast, the two-dimensional system realisations in Section 3.6 were all performed on a parallel supercomputer (details of such machines are given later in Section 5.2) in order that the much larger and more complex system can be visualised in very short time-scales — thus enabling interactive investigation of system parameter space. Full details of the parallel implementations of our models are given in Chapters 6 and 7; at this stage we intend to detail only the functionality of our programs.

3.1.1 Computer Implementation Details

The software written for this work is either standard C or FORTRAN code with some necessary additions to enable parallel execution. Our main design aim is to produce robust, flexible software that can be executed to model a variety of scenarios without the need for editing and re-compilation, and indeed, can be switched between modes of realisation at run-time. The user is therefore free to investigate system behaviour

without interruption. This functionality is provided through the use of command-line arguments, which can easily be stored in an input data file. The options available to the program user can be categorised as follows:

System description: The user can input specifications of the size and nature of the spatial system and the type of realisation to be executed. This involves providing the number of cell locations for each dimension of the system, as well as the number of species present. The realisation type can be specified as either deterministic or stochastic (see Chapter 5 for further details of these latter simulations), and as using either the full non-linear versions of the interaction equations, or alternatively, linear approximations. The user can also specify the length of a realisation by giving both the system time at which the run should terminate (all runs start with $t = 0$) and the time-step (δt) to be used.

Initial conditions: A small number of options are provided for the initial configuration of our system. All cells must start either empty of all reactants, or with equilibrium populations of all species; a simple *flag* is provided to *toggle* between these two options. The user can then specify some initial perturbation size and location. Although the most commonly used initial perturbation for our realisations is the simple addition of a single reactant to a single cell, options are available to perturb each reactant type by any given amount. In addition, initial perturbations can be made at a number of points within the spatial model, either contiguously or regularly distributed. This allows study of the transient interference between population movements caused by separate perturbations.

System parameters: Having described a particular physical system and the desired length and initial conditions of a realisation, the user must now specify the interaction and migration coefficients from the system equations. The program allows the user either to input each and every parameter individually, or alternatively to give values for the system instability (I) and the number of *waves* desired in the final steady-state solution. If the latter option is taken, the program assumes standard values for cross-interaction coefficients, based on Turing's case of "stationary waves with finite wavelength". The two given values are used to calculate self-interaction coefficients and migration rates, given the prior knowledge of the system size and dimensions. Full details of these calculations as well as those for

determining the relationship between linear and non-linear coefficients is given later in Section 3.1.2. An extra feature of our software is that a range of I-values can be requested, and given an increment value to step through this range, the program will execute a number of realisations, one for each successive value of system instability.

File output: Facilities are provided for the specification of two output files for each realisation. One file stores complete results from the run (i.e. all cell populations, input parameter listings, and rates of decay of populations); the other is reserved for recording values of global populations and the results of any special features requested by the user. Two of these extra options are provided by simple file output flags. One requests that only the final state of the system be recorded, and the other restricts the recording of data to a single user-specified cell, rather than for the complete system. Both of these options are useful to extract particular information from the very large data-sets that this program can produce.

Display options: As has been noted above, the user has the option of a graphical run-time display of reactant populations. This display takes the form of a set of circular graphs for the one-dimensional realisations, and a flat plane for the two-dimensional results. Populations values are shown by the height of the graph in the first case, and for the two-dimensional work by either colour-coding or (if working on a monochrome workstation) by displaying whether populations are above or below equilibrium values. Examples of all these graphical outputs can be viewed later on in this and other chapters. The program user has three simple options for use with the graphical display. First, the display can be activated with a toggle-flag; second, the number of displays to be taken from the run must be given (this therefore specifies the simulation time between displays); and finally an option is provided to enable reactant populations to be rescaled before display. This final option is particularly useful for the study of linear systems when populations can grow at an exponential rate.

Interrupt mode: If this option is requested then the program will pause after each display point to allow the user to change the nature of the realisation. This facility has proved very useful in testing the stability of steady-state solutions when they are subjected to further perturbations through adding or removing reactants from

particular cells during the course of the realisation. In addition, it has proved useful for the close investigation of the step-by-step evolution of the system. Within interrupt mode the user has the choice of the following options:

- add a perturbation of a given size to either a single or a number of cells;
- toggle between linear and non-linear versions of the system equations;
- toggle between deterministic realisation and stochastic simulation;
- save the current state of the system;
- adapt the step-size (δt);
- change the system instability (I); and
- cancel the interrupt mode and continue to the end of the realisation.

Special features: During the course of the development and use of the realisation code a number of extra features were added in order to investigate particular properties or behavioural modes of the system. The first of these changes the one-dimensional ring system into a line system by breaking the link between the cells opposite cell number one. No migration effects are allowed across this break, and the code is written so as to notify the user whenever the cell populations at the line-end move away from their initial state by a user-specified amount. This therefore provides notice of the point at which a line of N cells, with a perturbation at its mid-point, ceases to act in the same way as a ring of N cells with a single perturbation.

The realisation software also has the facility to record the velocity of perturbation effects moving through the system. This is achieved by the program recording the time that each cell's population moves away from its initial state by more than a given percentage of the equilibrium population. Recording the times at which this change first occurs in each successive cell gives a specification of the movement through the system of the effect of a perturbation.

Using a similar technique the program can be used to identify the realisation time at which the system has reached *stability*. Again the user specifies some percentage of equilibrium populations, and once the population movements of all cells within a single iteration are within this ceiling, stability is judged to exist and the time recorded. Details of the realisation results achieved with these special features are given later in this chapter.

Help information: It can be seen that the user of our software has a large number of options and facilities available. We have therefore provided a help option within the program. This displays all available command-line options, with explanations of their purpose and instructions on their use. Figure 3.1 gives a sample of the "help" option output, and thus also acts as a summary of all the functionality of our realisation software.

```
-a          (Adjacent cells initially perturbed)
-b          (Single cell initially perturbed)
-c  [int]   (Number of cells in ring)
-d  [file]  (Dump ascii values: give filename for totals)
-e          (Use equilibrium background populations initially)
-f  [file]  (Output file for full results)
-g  [float] (Set graphics on, give max population for graphs)
-h          (Help information - this screen)
-i  [float] (Minimum Instability)
-j  [float] (Instability Increment)
-k  [float] (Maximum Instability)
-l          (Use linear approximation)
-m          (Use Turing approximation)
-n          (Interrupt run to allow changes)
-o  [float] (Cut ring to form line, give change to spot arrival)
-p  [int]   (Number of data sets to be printed)
-q          (Set rescaling after each iteration)
-r  [int]   (Use stochastic simulation, give seed)
-S          (Save final state to file)
-s  [float] (Step-size for iteration loop)
-t  [float] (Finish time for run)
-u  [float] (Measure velocity, give percentage to recognise wave)
-v  [float] (Identify stability, give measure)
-w  [int]   (Number of waves around ring)
-x  [int]   (Initial X population perturbation)
-y  [int]   (Initial Y population perturbation)
-z  [int]   (Data dump limited to one cell, give number)
```

Figure 3.1: Help information from the standard realisation software
Running the standard realisation program with the "help" option (-h) produces screen output as in this figure. The screen therefore acts as a listing of all the available command-line options for the program. The first column gives the command-line option, the second specifies what (if any) additional parameter must be supplied, and the third column gives a brief description of option's effect.

3.1.2 Calculation of Realisation Parameters

Building on the analytic work detailed in Chapter 2 we require a mechanism for calculation of the interaction and migration coefficients to be used in our deterministic realisations. To do this we use the linear coefficients covered in Section 2.2 and the general non-linear interaction detailed in Section 2.4.2. We can thus convert an instability value and a desired number of waves into the appropriate interaction and migration coefficients to produce finite wavelength behaviour. At this stage we restrict ourselves to the one-dimensional two-reactant scenario; details for more complex cases are given later.

Interaction Coefficients

We can define inter-relations between linear and non-linear interaction coefficients by inserting the linearisation relations (2.2) into the non-linear equations (2.1) with interactions (2.19). We thus obtain

$$f(X_i, Y_i) = (X^* + x_i)[r_1 + a_1(X^* + x_i) + b_1(Y^* + y_i)] \quad \text{and}$$

$$g(X_i, Y_i) = (Y^* + y_i)[r_2 + a_2(X^* + x_i) + b_2(Y^* + y_i)].$$

By differentiating these equations with respect to the movements away from equilibrium (x_i and y_i), and then setting x_i and y_i to zero, we obtain at $X = X^*$ and $Y = Y^*$

$$\partial f/\partial x_i = r_1 + 2a_1X^* + b_1Y^* \quad \text{and} \quad \partial f/\partial y_i = b_1X^*,$$

$$\partial g/\partial x_i = a_2Y^* \quad \text{and} \quad \partial g/\partial y_i = r_2 + a_2X^* + 2b_2Y^* \ .$$

Equations (2.3) and (2.4) use the Taylor series expansion to give

$$a = \frac{\partial f(X^*, Y^*)}{\partial x_i}, \quad b = \frac{\partial f(X^*, Y^*)}{\partial y_i}, \quad c = \frac{\partial g(X^*, Y^*)}{\partial x_i} \quad \text{and} \quad d = \frac{\partial g(X^*, Y^*)}{\partial y_i} \ .$$

So we can equate the four linear coefficients to our six non-linear coefficients of the general quadratic scenario. Obtaining additional simplification from (2.19) we can write

down

$$a = a_1 X^*, \quad b = b_1 X^*, \quad c = a_2 Y^*, \quad \text{and} \quad d = b_2 Y^* \ . \tag{3.1}$$

These expressions, coupled with

$$r_1 + a_1 X^* + b_1 Y^* = 0 = r_2 + a_2 X^* + b_2 Y^*, \tag{3.2}$$

also derived from (2.19), allow us to convert from non-linear to linear coefficients.

It is often useful to employ Turing's simplification of reducing all the interaction coefficients to a single *Instability* value, I, and then defining some coefficients as constant and some as functions of I. The ability to concentrate on a single controlling parameter is most profitable, for example, during the investigation of possible parameter space (see Section 2.3.1). In such circumstances the above equations allow us to produce Turing's single linear instability value from the corresponding full non-linear system.

Migration Rates

In the course of studying our realisations we are often not particularly concerned with studying the system behaviour given some particular value of migration rate for each species. We are more often interested in studying the behaviour of specific wave-like solutions to our system, and so desire that migration rates be set to the particular values that will produce such behaviour. We therefore turn to our analyses of the previous chapter (in particular Section 2.2.1) to define the migration rates μ and ν in terms of a single variable (U) along with the size of the system (N) and the number of waves (s_0) present in the asymptotic solution, i.e. re-working (2.7) produces

$$U = 4\mu \sin^2(\pi s_0/N) = 4m\nu \sin^2(\pi s_0/N),$$

where m is the ratio of the migration rates (μ/ν). Using this simplification we can write equation (2.7) as

$$(p - I)^2 + (\frac{1}{2} + \frac{3U}{2})(p - I) + \frac{1}{2}(U - \frac{1}{2})^2 = 0 \ . \tag{3.3}$$

If $U = 1/2$ then (3.3) has the straightforward single solution equating to the value of instability, $p = I$. We will therefore often choose this value for U during our investigations, thereby allowing the migration rates to be determined from the user-requested value of s_0.

3.1.3 The General Non-Linear Interaction Coefficients

Although this work covers a wide variety of spatial reaction systems, we will often return to one particular scenario for reference, comparison and further investigation. This case corresponds to the non-linear version of Turing's system of "stationary waves of finite wavelength", and has produced a wealth of interesting results that are detailed later in this chapter. In particular we are interested in extending the work of Renshaw [1991, cf. page 321]. In this he developed interactions that produced dynamic wave patterns for stochastic simulations with non-linear interactions: we now study both the general stochastic and deterministic schemes, and investigate the behaviour of systems that exhibit permanent wave structures.

Turing provides us with a particular set of parameters that can be used to produce stationary waves of finite wave-length (see Turing [1952], page 52); these are $a = I - 2$, $b = 2.5$, $c = -1.25$, $d = I + 1.5$, $m = 2$ and $U = 1/2$. The one-dimensional systems that we investigate are typically chosen to contain 50 cells, with five complete wave structures present in the final steady-state solutions. Such systems are of sufficient size to allow the production of large-scale spatial effects, but can also be realised in short timescales (i.e. a few minutes) to enable study of many system parameters. With equilibrium reactant populations chosen as $X^* = Y^* = 10$ say, from equations (3.3), (3.2) and (3.1) we can determine a set of non-linear coefficients that satisfy the given the instability value I. For example, Table 3.1 details the coefficients that are used when we choose $I = -0.2$ (e.g. as we do in the final example in Section 3.3.2).

3.2 General One-Dimensional Realisation Results

The software detailed in the previous section enables us to perform a broad-ranging investigation into the behaviour of deterministic one-dimensional spatial reaction sys-

Migration coefficients	μ	=	1.3090
	ν	=	0.6545
Interaction coefficients	a_1	=	-0.220
	a_2	=	-0.125
	b_1	=	0.250
	b_2	=	0.130
	r_1	=	-0.300
	r_2	=	-0.050

Table 3.1: Standard simulation system parameters
The choice of these parameters will produce sub-critical behaviour in a two-reactant one-dimensional spatial reaction system, with instability $I = -0.2$. These values are taken directly from Renshaw [1991].

tems. We first investigate the types of behaviour that Turing predicted for this system with linearised interactions and periodic boundary conditions. Although the majority of these cases are limiting in terms of certain parameters (and therefore represent a small proportion of the available parameter space) they do show some specific types of behaviour. The majority of these studies confirm Turing's predictions, although there are a number of scenarios in which the systems develop in a manner more complex than previously analysed.

The predominant part of the results within this chapter are concerned with one of Turing's cases in particular — that of stationary waves of finite wavelength. Not only is this case the most relevant in terms of representing physical or chemical systems, but as described in Chapter 2 it also accounts for the vast majority of the available parameter space. Our studies of this case include:

- an investigation into the effect of varying system instability I and other parameters;
- an examination of the *morphological stability* produced in linear scenarios where the actual reactant populations are growing or decaying exponentially;
- a comparison of the development of the non-linear and linear systems, including a description of the circumstances where we can regard the linear approximation as accurate;
- a study into the rates of decay of perturbations back to steady-state solutions, and

how these decay rates are related to the instability of the interactions;

- an analysis of the stability of wave structures as initial perturbations and I-values are varied — this leads us to an empirical specification of when wave structure breaks down;

- a study of the number of waves that appear on a ring when migration rates are held constant, but the number of cells, N, is varied.

This final item has produced significant results in terms of the importance of using a discrete-space or stepping-stone model rather than a diffusion-based realisation. We have observed certain interesting, and potentially physically relevant, effects on our discrete system that cannot be reproduced by continuous models of the same system. In particular we report on a study of the effects of using very small ring systems, and the way in which the wave structures then map onto the physical system. A categorisation of wave types is introduced and used to map the wave structures produced as ring size and instability are varied.

We will concentrate on the finite wavelength system later in Section 3.3. In advance of this, however, let us examine the results of our realisations for the two limiting case scenarios — those with extreme-long and extreme-short wavelength wave patterns. In Section 2.2.3 we detailed the possible behavioural cases for the one-dimensional two-reactant spatial reaction system with linear interactions. We now present the results of our computer realisations of these linearised systems to enhance these earlier descriptions. We are thus constrained to vary only the reactant migration rates (μ and ν), the cross-interaction coefficients (b and c), and the self-interaction coefficients (a and d).

3.2.1 Extreme-Long Wavelength Scenarios

In this scenario the wavelength of reactant populations is so long that it encompasses all cells in the system. In effect, we therefore see unified movement of all cell populations, either away from, or towards, equilibrium. This case occurs when we restrict reactant migration rates and self-interaction coefficients to be equal, and the cross-interaction coefficients to have $|b| = |c| = 1$. If these cross-coefficients are both positive or negative

(i.e. $bc > 0$) we have linear stationary waves with extreme-long wavelength, if one is negative (and thus $bc < 0$) we obtain the oscillatory kilter (see Section 2.2.3).

Turing states that for both the stationary and oscillatory kilter the linear system is in unstable equilibrium, and thus any perturbation will create an explosive movement in the reactant populations. In the stationary case this movement is constantly in one direction, whilst in the oscillatory kilter the movement has a periodically changing sign. Our realisations confirm these predictions, but only if we ensure that the self-interaction coefficients, a and d, remain positive. In the linearised stationary kilter all cells move away from equilibrium with amplitudes of the same sign as the initial perturbation. In the oscillatory case the populations oscillate (see Figure 3.2), and changing the sign of the perturbation produces the same development, but with the reactants reversed.

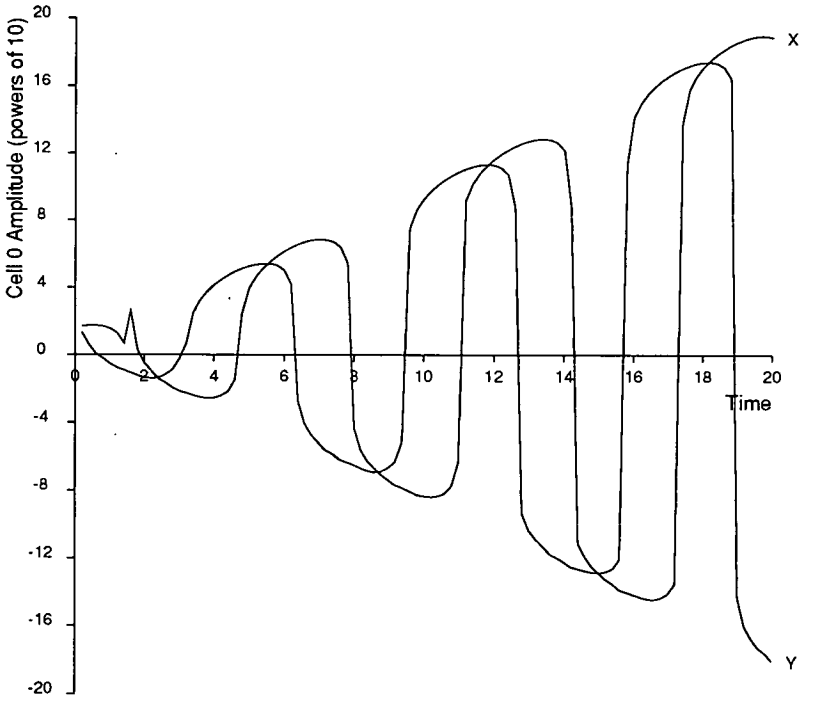

Figure 3.2: Extreme-long wavelength reactant populations

A graph of the reactant populations of a perturbed cell in a one-dimensional, two-reactant spatial reaction system with linear interaction coefficients chosen to produce oscillatory, extreme-long wavelength behaviour. It can be seen that the two populations oscillate out of phase by $\pi/2$, once the transient effects of the perturbations has passed.

However, if we consider the situation with $a = d = -1$, we observe very different behaviour to that detailed above. Our realisations now all produce an inherently stable linear system, with all perturbations decaying back to equilibrium, rather than exploding.

We still observe the fundamentals of extreme-long wavelength behaviour, but with exponential decay to equilibrium, and with $bc < 0$ we still obtain amplitudes that change sign periodically. However, with negative self-interaction coefficients our results definitely show that the linear system is stable to perturbations. This is our first empirical experience of the need to consider system criticality as well as kilter. With $a = d = -1$ we have moved into a sub-critical scenario, whereas using Turing's values the system behaves super-critically

We have also investigated running systems with non-linear interactions equivalent to the linear ones detailed above. In the super-critical case with $a = d = 1$ the systems behave very much as the linear case provided the initial perturbation is positive, the only difference being that the population explosion is more severe. In fact for the oscillatory kilter, the explosion can rise in one direction so rapidly that the system undergoes no oscillations at all. However, for the unstable non-linear system with negative initial perturbations the reactant populations decrease exponentially towards zero, hence reaching a permanent stable (if "uninteresting") state.

In the inherently stable (sub-critical) extreme-long wavelength scenario ($a = d = -1$) we observe behaviour exactly like the linear case, with all perturbations returning to the equilibrium position. However, the non-linear interactions are much more sensitive to perturbation size. We find that should initial perturbations in one population exceed 10% of equilibrium population level, then the system is in danger of being overwhelmed by transient activity, and a localised irreversible population explosion can occur. Thus even when choosing parameters to give a sub-critical non-linear system, initial conditions (and system discreteness) can produce inherent local instability that dominates the final solution.

3.2.2 Extreme-Short Wavelength Scenarios

In this case of limiting behaviour we see the first true wave-like structures appearing in our realisations. Extreme-short wavelength development corresponds to each cell in the one-dimensional system behaving in a manner opposite to that of its neighbours. We can thus obtain spatial population distributions with a periodic structure. We can produce the stationary kilter (the oscillatory kilter will be discussed later in Section 3.5)

of this type of system by setting one of the reactants migration rates to be zero, and by restricting $bc < 0$. It is important to note (again see Section 2.2.3) that unlike the extreme-long wavelength case, stationary behaviour requires bc to be negative rather than positive. In this scenario we can also introduce Turing's simplifying instability parameter, I, and thus define the self-interaction coefficients as $a = I - 1$ and $d = I$. Given $b = -c = 1$, $\mu = 1$ and $\nu = 0$, this allowed Turing to write the solution of equation (2.7) as

$$p_s = I - 1/2 - 2\sin^2(\pi s/N) + \sqrt{(2\sin^2(\pi s/N) + 1/2)^2 - 1} \ . \qquad (3.4)$$

The largest possible value of this solution (p_{s_0}) occurs when the $\sin^2(\pi s/N)$ term is greatest, i.e. when $s/N = 1/2$. The dominant asymptotic solution of the linear system should therefore contain $N/2$ waves evenly distributed around the ring – this, of course, fits ideally with adjacent cells acting alternately as crests and troughs in the wave structure, and this result is confirmed by our realisations. The resulting structure of the linearised system is morphologically similar to that shown in Figure 3.3, but with reactant populations increasing or decreasing exponentially in time.

We have performed a number of realisations of the above one-dimensional system with two reactants, and an investigation of the system with varying instability values has proved to be of particular interest. As expected, if I is chosen to be negative (Turing does not consider this sub-critical option), any perturbations to a system result in decay back to the steady-state equilibrium case; this occurs for realisations with both linear and non-linear interaction functions. For super-critical systems we expect any perturbation to result in extreme-short wavelength waves appearing throughout the system, and thus reactant populations to grow exponentially in magnitude, with a high correlation between next-nearest neighbour cells. For this system with linear interactions, we find that for $I = 0.1$ and $I = 0.2$ the reactant populations returned to equilibrium (showing sub-critical behaviour), although with I increased to 0.3, permanent short-wavelength waves are produced. If we set $s/N = 1/2$ in Equation (2.7) we obtain

$$p_{s_0} = I - (\sqrt{21} - 5)/2 = I - 0.20871 \ ,$$

and this reveals that for the leading eigenvalue of the linear solution to be positive (thereby giving true instability or super-criticality) we must have $I > 0.20871$. We

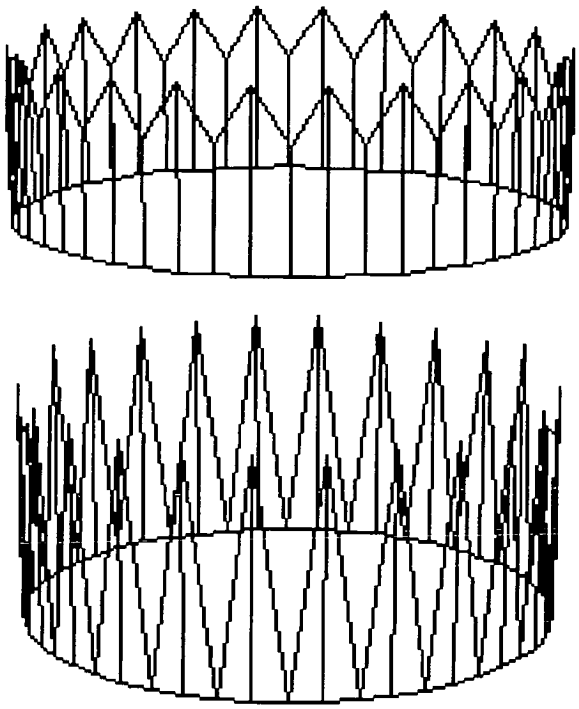

Figure 3.3: Extreme-short wavelength reactant populations
A graphical example of the reactant population waves for the case of extreme-short wavelength in a one-dimensional implementation of a two-reactant spatial reaction system with non-linear interactions. Migration rates have been fixed at $\mu = 1$ and $\nu = 0$, cross-interaction coefficients are $b = -c = 1$, self-interaction coefficients are $a = I - 1$ and $d = I$, and I has been chosen as 0.3 for this realisation. Note the stable wave structures with differing amplitudes for each reactant; no rescaling has been used for this experiment. The upper graph represents the X-reactant, and the lower graph the Y-reactant.

therefore see that in the extreme-short wavelength scenario the sign of Turing's (finite wavelength) instability value does not provide the divide between system criticalities in the extreme behaviour cases. Thus even with a positive I-value as proposed by Turing, we can obtain sub-critical activity that is not described in his work. We must therefore look to Equation (2.7) to find our criticality divide.

This same modification to the threshold I-value holds true for our empirical investigation of the equivalent system with non-linear interactions. For example, with $I = 0.3$, we obtain a ring of short-wavelength waves which have a stable fixed amplitude. This is our first example of permanent stable wave structures, Figure 3.3 provides an illustration of such a system.

Our investigations then move on to non-linear systems with higher instability values, and this produces additional interesting results. For example, with $I = 0.5$ we obtain a

system that exhibits almost chaotic behaviour in terms of the amplitudes and locations of waves. The extra instability seems to drive several modes of spatial oscillation around the ring, and it thus becomes difficult for the largest eigenvalue solution to dominate. A few snap-shots of the reactant population for such a system are shown in Figure 3.4, where we can observe the changing wave patterns. This type of model is particularly interesting as such behaviour may be a lot closer to natural systems than some of the more simplistic scenarios we have previously observed. As we increase the instability further, this chaotic behaviour becomes more extreme, until, with $I = 1.0$ sudden population explosions in one species lead to extinction of both reactants, and this extinction then spreads around the ring to reach a final, empty steady-state.

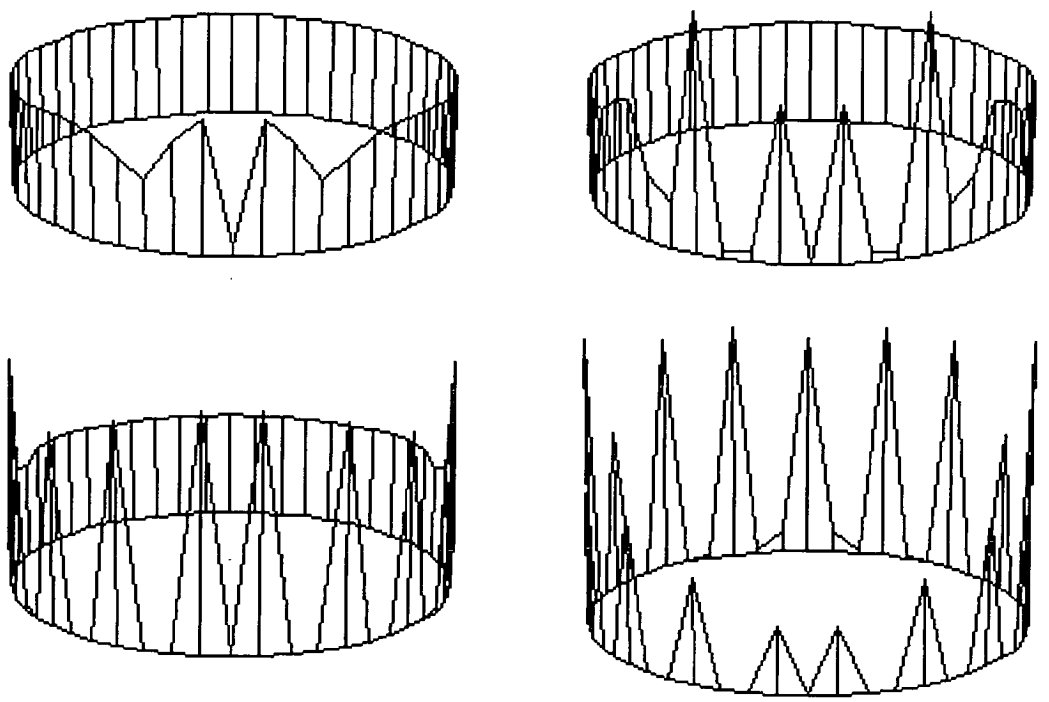

Figure 3.4: Complex patterns in the extreme-short wavelength scenario
Time evolution of the population of reactant Y in a one-dimensional two-reactant spatial reaction system with non-linear interactions and a single initial perturbation. Coefficients are chosen to produce extreme-short wavelength behaviour, but the chaotic nature of the development is produced by using a relatively high instability parameter of $I = 0.5$. The four graphs represent the reactant populations at times $t = 10.0$ (top left), $t = 26.0$ (top right), $t = 50.0$ (bottom left) and lastly $t = 95.0$ (bottom right).

Our experiments with the stationary extreme-short wavelength scenario are concerned with systems where the *natural* wave structure is incompatible with the ring structure. The simplest example of this is to consider a ring with an odd number of cells, since

we cannot now have a regular set of $N/2$ waves, each with a wavelength of two cells. We have found that, providing I is large enough to create genuine instability, Turing's claim of regularly varying wave amplitudes around the ring is correct for the linear case. However, with non-linear interaction functions we find that all wave amplitudes are identical, except for the point opposite the initial perturbation. Here the spatial structure of the pattern adapts to account for the *missing* cell, and thus a cut-off wave is formed. We therefore have $N/2 - 1$ waves, all but one of which have identical amplitudes (see Figure 3.5). This behaviour is interesting because the non-linear short-wavelength reaction we see here is very similar to the linear finite wavelength behaviour that we will study in Section 3.3. It also provides us with an example of behaviour that is significantly different between systems with linear and non-linear interactions, and which cannot be produced using diffusion-based continuous systems.

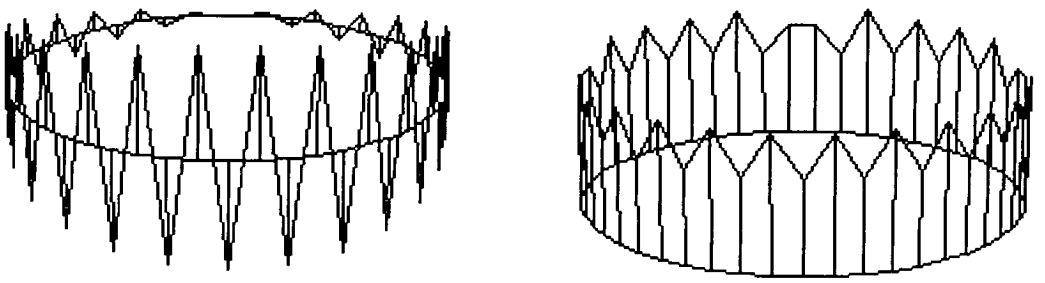

Figure 3.5: Extreme-short wavelength waves on an odd number of cells
A comparison of the linear and non-linear reaction to an extreme-short wavelength two-reactant scenario with an odd number of cells. In this case we have $N = 49$, and the left hand graph shows the rescaled variation of reactant X with linear interactions, and the right hand graph shows X with equivalent non-linear interactions. Both graphs are taken at time $t = 800$, and equate to explosive linear amplitudes of over 10^{32} at the point of perturbation (the front of the ring as viewed), and bounded, spatially constant, non-linear amplitudes. This massive contrast in realisation populations is a fundamental difference between linear and non-linear systems.

3.3 Finite Wavelength Scenario Behaviour

Let us now move on to our main area of interest — the production of wave structures with a *finite* wavelength, where we use Turing's definition of finite, referring to all system scenarios other than those with "extreme" wavelength solutions. In Section 2.2.3 we detailed linear coefficients to produce such behaviour, and in Section 2.3.1 we discussed the regions of parameter space within which this behaviour occurs. Using this work

as a base, we can now complete an extensive investigation into the production of finite wavelength patterns in both the linear and non-linear scenarios.

For the majority of the results in this section, we have used Turing's suggested migration and interaction coefficients for stationary waves, using the techniques presented in Section 3.1.2 to convert to non-linear interaction coefficients where necessary. Adopting this approach allows us to unify representation of the interaction coefficients into just a single variable (the instability parameter I), and results in the migration rates being determined by the values chosen for the ring size, N, and the number of desired waves, s. The actual coefficient values used for the majority of realisations are therefore as detailed in Section 3.1.3 for the linear interaction functions, and in Table 3.1 for non-linear interactions. This choice produces stationary kilter wave patterns in the linear scenario, provided we use a positive value for I — a decision that is implicit to all of Turing's results. The amplitude of these waves grows rapidly, and is appears sensible to regard the structures in terms of their morphological stability only. Since for any natural system such exponential population or concentration growths would always reach some limiting factor not accounted for by the linearised interaction functions, hence rendering the linearisation assumptions invalid.

We thus concentrate a large proportion of our studies on investigating more physically realistic behaviour by using non-linear interaction functions, thereby building a portfolio of comparisons between the linear and non-linear variants. In particular we have found that non-linear systems can produce finite stationary wave patterns with a constrained amplitude, and we provide an analysis of such systems in terms of their variation with initial conditions and system instability.

A second important study concerns realisations where I is not restricted to be positive. We detail realisation results to support our proposal that I be considered as a measure of the criticality of the system. Thus positive I values produce *super-critical* behaviour (i.e. exponentially exploding wave amplitudes in the linear scenario, and permanent waves with non-linear interaction functions); negative I values produce *sub-critical* activity (where we expect any perturbations to return to stable equilibrium); and with $I = 0$ we obtain the *critical* state, where behaviour proves to be more complex, and far more dependent upon initial conditions.

3.3.1 A Comparison of Linear and Non-linear Systems

The most important result of these realisations is the confirmation that, by choosing parameters as suggested by Turing, we can reproduce spatial distributions of reactants with wave-like patterns. This is of great relevance to scientists working with spatial reaction systems in many fields, since the ability to analyse and simulate systems without boundary conditions becomes possible. Provided we choose interaction coefficients and a migration rate ratio that lie in the relevant section of possible parameter space (see Section 2.3.1), it is possible to adjust the magnitude of reactant migration rates to produce reactant population distributions with periodic variations of a specific wavelength. Figure 3.6 shows the typical wave structures produced by both linear and non-linear interactions, when migration rates are chosen to produce five waves around the ring system.

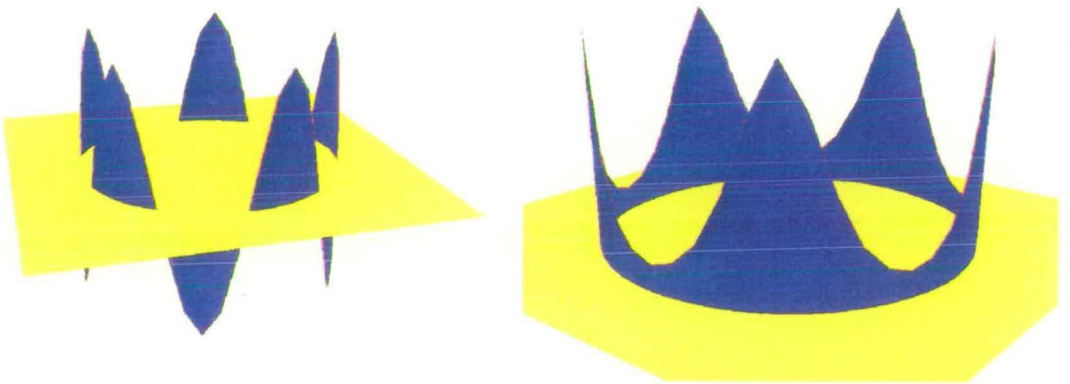

Figure 3.6: Linear and non-linear wave structures on a ring of cells
Circular graphs of the population of reactant X in a one-dimensional two-reactant spatial reaction system containing 50 cells. Interaction coefficients and migration rates are chosen to give five waves. The yellow plane in the diagrams represents zero populations, it can been seen that the linear realisation (left) contains negative populations. In addition, this graph is rescaled according to the maximum population, since this explodes exponentially with time. Conversely, the non-linear graph (right) is not rescaled, and highlights that these wave structures produce stable waves with positive population values.

In general, wave structures of this type are produced in our ring systems for any displacement of the system from its equilibrium state. If we have genuine instability in our interactions (i.e. a super-critical system, $I > 0$) the effect of any perturbation will spread throughout the ring to produce persistent waves. This super-critical case is detailed later in Section 3.3.2, and the effect of perturbation spread is shown below

in Figure 3.7. This same effect is observed for other possible instability values in the few time periods after realisations begin (say, $t < 25$); though for cases where $I \leq 0$ we do not obtain distinct permanent waves. The sub-critical scenario (with $I < 0$) is straightforward — although we can observe transient wave-like structures, with the required wavelength, all perturbations to the system are attenuated and the final system state returns to equilibrium. We will discuss this decay later in Section 3.3.3. The critical case, with $I = 0$, is more complex. Here we have no leading eigenvalue for the linearised solutions, and our experiments show that system development does produce wave structures similar to the sub- and super-critical cases, but the asymptotic state of the system is highly dependent on initial conditions.

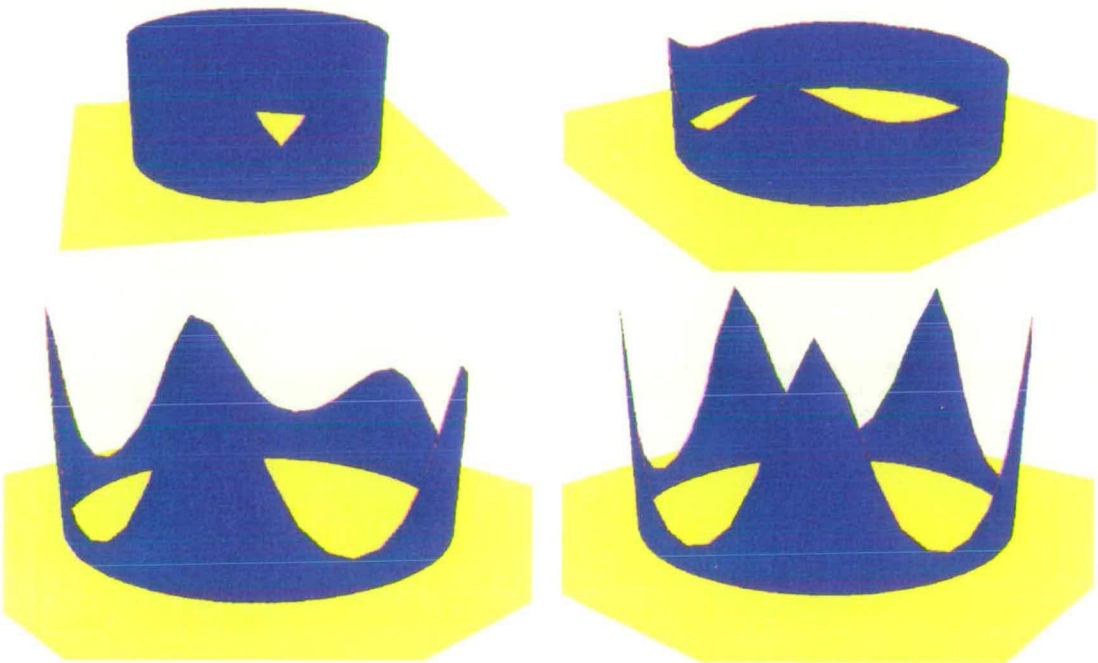

Figure 3.7: Permanent wave development from a single perturbation
These four graphs show the spread of influence of a single perturbation through a two-reactant spatial reaction system on a ring. The graphs show the population of reactant X at times $t = 0$ (top left), $t = 2.0$ (top right), $t = 20.0$ (bottom left), and $t = 500$ (bottom right). The system shown contains 50 cells, and non-linear interaction functions are used. The graphs are taken from a graphical display in which the position of the initial perturbation rotates in time.

Later in this chapter we will discuss matters of the discreteness of our realisations, the stability of our models, and the effects of varying system instability and initial conditions. It is important to understand how the non-linear versions of these systems differ from those with linear interaction functions, for which we have been able to

perform the most complete mathematical analysis. Figure 3.6, and the brief discussion above, show that there are quite substantial differences between the development of linear and non-linear super-critical scenarios, although the morphological nature of the waves is roughly similar. However, we obtain much closer agreement between linear and non-linear realisations for sub-critical runs. Although we observe significant difference in transient activity as the systems move closer to their equilibrium states, we see a convergence in both total system population and wave amplitude decay rates (see Section 3.3.3).

We can compare total ring populations and their variation with time for linear and non-linear sub-critical systems ($I = -0.2$), with two types of initial conditions. The first of these, described as the *small* perturbation system, consists of all cells with equilibrium populations ($X^* = Y^* = 10$), except for five cells (evenly distributed around the ring) in which both X and Y populations are perturbed by a small positive amount (six units). In the second case, with *large* perturbations, all cells are initially empty, but five cells have X and Y populations just in excess of the equilibrium levels. Figure 3.8 shows the time development of total ring populations for both these cases. We observe that the same final total population is consistent between linear and non-linear realisations. Where differences do occur, they are in the transient phase (i.e. with $t < 10$ for small perturbations, and $t < 20$ for large perturbations) when the higher-order terms in the non-linear interactions take effect. Once all perturbations are small, the linearisation approximation proves accurate. It is interesting to note, however, the difference in the way the two realisations approach equilibrium in the large perturbation case. The system with linear interactions does so in a "convex" manner, i.e. the second time-differential of the population is generally negative during the transient phase, without a change in the sign of the gradient of the line between time $t = 1$ and equilibrium being reached ($t \simeq 15$). In contrast, the non-linear system approaches equilibrium in a "concave" manner with at least three oscillations in the approach, but with the second differential of the population being generally positive. The linearised system is therefore acting similar to an *over-damped* version of the non-linear system.

The final case for this comparison of linear and non-linear systems is for system behaviour at criticality ($I = 0$). Our empirical results show that under these conditions our systems can exhibit very striking behaviour that is highly dependent upon initial conditions. However, the persistent structure will always lie close to equilibrium (un-

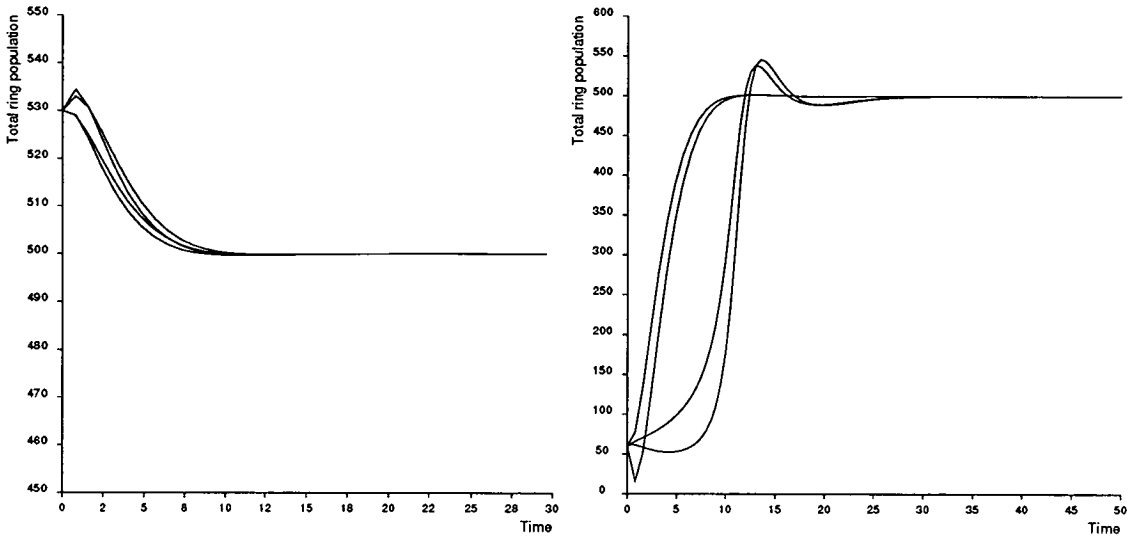

Figure 3.8: Sub-critical systems approaching equilibrium

Total populations on a 50-cell two-reactant spatial reaction system with $I = -0.2$, showing the difference between non-linear and linear numerical solutions. The left hand graphs shows *small* perturbations returning to equilibrium, the upper two lines represent the X and Y populations for non-linear interactions, the lower represents the linear equivalent. The right hand graph shows the same system, but with *large* perturbations. The higher-order terms in the non-linear interactions can be seen to affect the transient phase of system activity, and produce damped oscillations in the total populations of X and Y. In all cases, however, ring populations return to equilibrium. Note the different scales on these two graphs.

less initial perturbations are excessively large) and thus the linear critical systems are generally an accurate approximation to the non-linear system. At criticality, since there is no dominating eigenvalue for our solution, persistent system behaviour is very slow to develop. This presents us with a problem when realising the system on computer, since we often find it difficult to determine when a final steady-state solution has been achieved. We feel that this subject is important enough, and our results are interesting enough to warrant specific treatment, and therefore we will return to these realisations later in Section 3.3.4.

3.3.2 Super-Critical Non-Linear Behaviour — Permanent Waves

As shown earlier, the correct choice of system parameters can produce persistent, stable wave structures in non-linear realisations of two-reactant, one-dimensional spatial reaction systems. In the course of our empirical studies we have discovered some

I	Perturbation Type (see text) Multi-Extreme	Single-Extreme	Multi-Small	Single-Small
-0.15	Decay — Bounce — Decay to Equil.	Decay — Bounce — O-waves	Decay to Equil.	Decay to Equil.
-0.10	Decay — Explode — Decay to Equil.	Decay — Explode — O-waves	Decay to Equil.	Decay to Equil.
-0.05	Decay to Zero	Decay to Zero	Decay to Equil.	Decay to Equil.
0.0	Decay to Zero	Decay to Zero	Small "stable" waves	Growing then decaying waves
0.05	Decay to Zero	Decay to Zero	Stable waves	Stable waves
0.10	Decay to Zero	Decay to Zero	Stable waves	Stable waves

Table 3.2: Wave behaviour patterns in non-linear systems

Descriptions are given of the wave pattern development for our generic spatial reaction system with four types of initial conditions, and a range of instability values. In general the table describes wave structures that "Grow" or "Decay", with "Explode" referring to very rapid growth. "Bounce" refers to a period of rapid growth in wave amplitude, but followed by a return to stable activity. "O-waves" are travelling waves of reactant as introduced in Chapter 2. The various other terms used to describe the perturbation types and the different wave patterns are detailed in the surrounding text.

interesting types of system behaviour, and some unusual special case scenarios. In Table 3.2 we summarise the wave structure behaviour of our generic ring system, as we experiment with different initial conditions and move through from sub-critical to super-critical states. Besides providing an overview of empirical system development, this table also allows us to identify *inflection points*, where wave structure formation patterns change from one type to another. In one case this point separates systems that have a zero-population final state from those with a (non-zero) equilibrium population final state (with an explosive interface between the two), and in another case the table serves to highlight the criticality boundary.

In general, our realisation experiments used four different sets of initial conditions. Some of these have been mentioned earlier; let us now detail these types more exactly.

Multi-Extreme: All cells are initially empty, except that a given number of cells (regularly distributed throughout the spatial domain) are populated with both reactants to a level 20% above the equilibrium cell population. The number of cells to be perturbed is normally taken to be equal to the desired number of

wavelengths around the ring.

Single-Extreme: Again all cells are initially empty of reactants, but now just a single cell (index 0) is perturbed with a population 20% above the expected equilibrium value.

Multi-Small: All cells are now populated with equilibrium reactant populations, and as such produce a steady-state non-zero solution. However, under this perturbation we add 20% of the equilibrium populations to both reactants in a given number of cells. The number of perturbed cells is (by default) equal to the expected number of waves, and these perturbed cells are distributed evenly around the ring.

Single-Small: Like the previous case we now have all cells populated with equilibrium levels of reactants. A single cell (index 0) is then perturbed by adding 20% to both reactant populations.

It is clear from Table 3.2 that for the case of *small* perturbations, the criticality point ($I = 0$) acts as the dividing line between steady-state equilibrium populations and stable wave patterns. The amplitude of the waves observed in these realisations is related to the magnitude of I, and is totally independent of the size of initial perturbations (providing they are not extreme). Figure 3.9 shows this variation in maximum cell population (at final stability) with increasing instability. It can be seen that following an initial rise in peak populations, these level-off above $I = 0.3$, and then decrease as we increase I even further. Once we move above $I = 0.5$ all realisations become unstable in their persistent state. The results shown in Figure 3.9 are for the *Multi-Small* initial conditions, although the *Single-Small* scenario produces almost identical amplitude results.

When we consider high values (i.e. $I > 0.3$) of instability in the super-critical case we find aberrations in the expected system morphology. Once the system instability value is above $I = 0.3$ the final persistent waveform is spatially distorted, and the wave amplitude is reduced. Figure 3.10 shows a set of wave patterns for various super-critical instability values. It can be seen that for the higher instability values the location of the first and fifth wave-crest has been displaced by one cell towards the cell that contained the initial perturbation (cell 0). In addition, when the instability reaches $I = 0.5$ we lose the wave-crest opposite the initial perturbation, and are thus left with only four complete waves. These effects are independent of the numerical technique used and

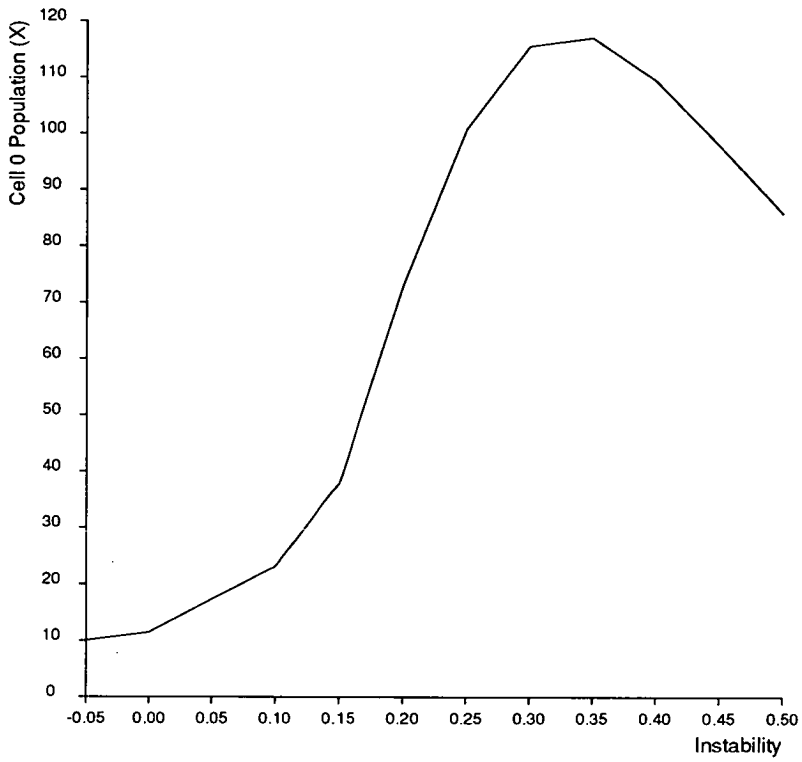

Figure 3.9: Variation of wave amplitude with interaction instability
The population in cell 0 of a two-reactant ring reaction system is recorded once the system has reached a persistent steady-state solution (i.e. at time $t = 200.0$). All the realisations performed for these results used the *Multi-Small* initial conditions and our generic interaction and migration coefficients on a ring of 50 cells.

highlight the importance of considering discrete rather than diffusion models, especially for systems of high instability. We will discuss the importance of discreteness further in Section 3.3.4.

The second interesting effect apparent in Table 3.2 is that produced by using extreme perturbations. All such cases with positive instability, i.e. super-critical realisations with non-linear interaction functions, do not in fact produce the expected wave patterns, but result in final states with zero reactants in all cells. This effect is independent of whether a single, or several, cells are perturbed. It seems that when *real* instability exists, large perturbations have such a strong effect that they prevent the leading eigenvalue solution from having its expected persistent effect. What is even more interesting is that extreme perturbations can also affect non-linear systems at criticality, and sub-criticality when I is small in magnitude. Only when we reduce the instability to $I = -0.1$ and below do we achieve the same final system state as with the small perturbation realisations, and this final state is only reached after some unusual transient behaviour.

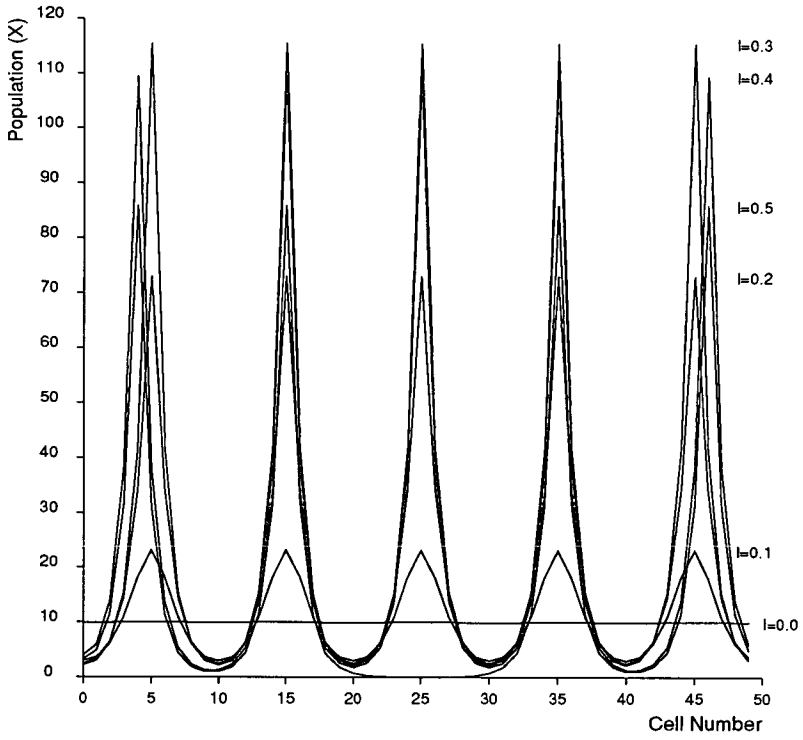

Figure 3.10: Wave crest displacement at high instability values

Reactant populations for a two-reactant ring reaction system, recorded once the system has reached a persistent steady-state solution (i.e. at time $t = 200.0$). All the realisations performed for these results used the *Single-Small* initial conditions with a negative perturbation, and our generic interaction and migration coefficients on a ring of 50 cells. Note the displacement of two wave crests for high I values, and the complete loss of one crest when $I = 0.5$.

As is recorded in Table 3.2, with $I = -0.15$ the extreme perturbations (*single* and *multi*) undergo an initial decay period, and the reactant populations can reach surprisingly low levels as they spread around the ring (see Figure 3.11). At a certain simulation time (dependent upon the instability of the system), namely around $t = 35.0$ when $I = -0.1$ (see Figure 3.11) and close to $t = 12.0$ with $I = -0.2$ (as in Figure 3.12), these low populations *bounce*. That is we observe either wave structures with the expected number of waves, and with a sudden increase in amplitude for the *multi* perturbation case; or we observe the gradual spread of a single wavefront around the ring for the *single* perturbation case. We have termed this single wavefront spread as *O-waves*, and these are described in more detail later in Section 3.4. For the multiple perturbation (*bounce*) scenario the amplitude of the waves produced then gradually decays, and the system returns to the expected sub-critical state with all cells having equilibrium populations of both reactants. Figure 3.12 shows the four main stages of this *Multi-Extreme* sub-critical

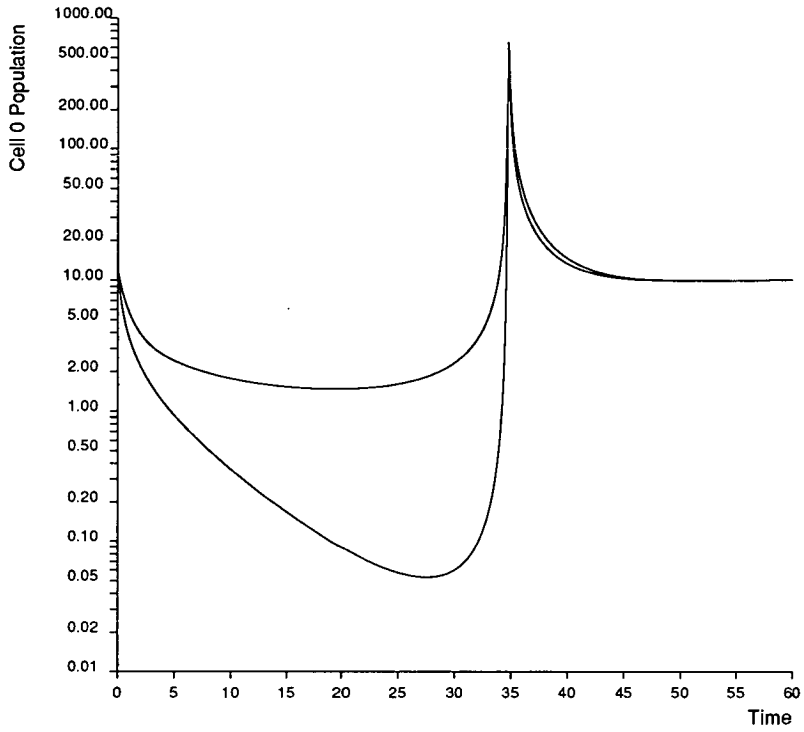

Figure 3.11: Cell populations in an explosive sub-critical realisation
The X and Y populations of cell 0 from a sub-critical spatial reaction system in one-dimension, with two reactants, non-linear interactions ($I = -0.1$), and *multi-extreme* initial conditions. Note that the population scale is logarithmic, and that the X-reactant curve is the lower of the two during the initial decay phase and the upper of the two during the post-explosion decay.

system development.

It is interesting to note the strong wave structures that are produced by this sub-critical system in the short period before these decay back to equilibrium. There is a substantial shift of all cell populations between times $t = 2.0$ and $t = 12.0$ (see Figure 3.12 for the case with $I = -0.2$). They move from small amplitude waves with reactant populations values of one to two per cell (i.e. approximately the initial perturbations redistributed among all cells), to large amplitude structures with approximately equilibrium populations in all cells. It is only from this point that the sub-critical decay behaviour dominates, and the wave amplitudes gradually decrease to produce the final steady-state equilibrium solution.

As we increase our system instability to $I = -0.1$, we observe a similar pattern of behaviour for both types of extreme initial conditions. However, the initial decay period is followed by a far more violent rise in reactant populations. We have therefore termed this rise an *explosion* rather than a *bounce*, since not only do the populations rise very rapidly, but they also reach very high values, before either forming stationary waves of decaying amplitude or travelling O-waves (see Section 2.4.2). Figure 3.11 shows the development in time of the reactant populations in cell 0 during a realisation with multi-extreme initial conditions. It should be noted that the population scale on this graph is logarithmic, and the extremes of the X-reactant population in particular are clearly visible — the population of the perturbed cell rises by over four orders of magnitude within eight time units. This explosive rise in populations becomes more severe as we increase the interaction instability in our realisation, and becomes so severe after a particular threshold that we lose all persistent behaviour other than the absence of any reactant. Table 3.2 shows this effect as the *extreme* initial condition scenarios "Decay to Zero" when $I > -0.10$. A closer investigation of the multi-extreme case shows this threshold to lie at $I \simeq -0.093$, with its exact position dependent upon the choice of perturbation size and temporal discreteness (δt) used in the realisation. The important result here is that, in the non-linear domain, strong initial perturbations can force a system out of its "expected" behaviour in all the criticality scenarios. We can thus observe very regular, large amplitude transient wave structures in sub-critical systems before the standard decay behaviour dominates the asymptotic result. We detail this same behaviour later in our stochastic simulations (Chapter 5), and show that it can have a very significant bearing on the effectiveness and efficiency of naïve computer

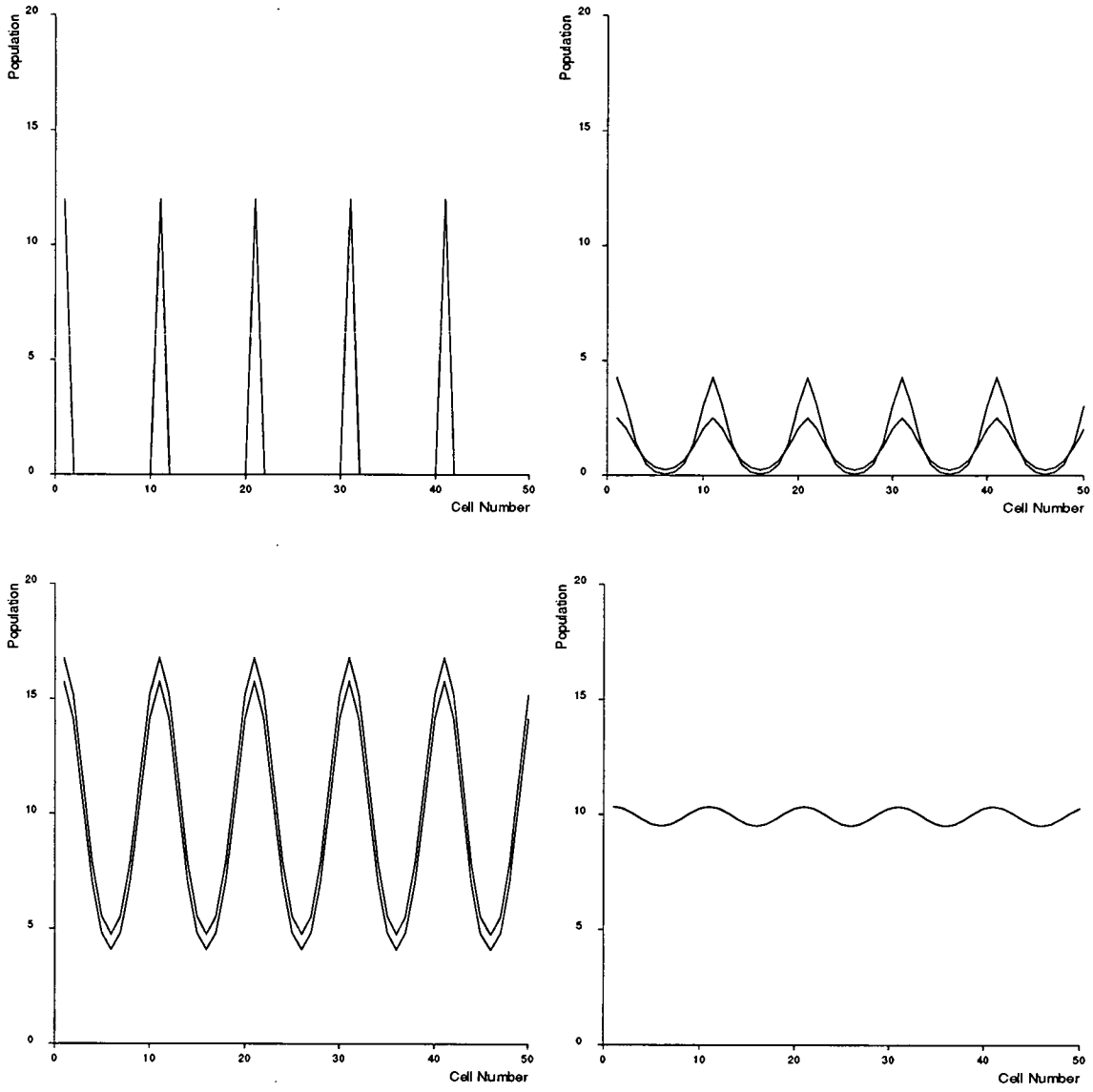

Figure 3.12: Non-linear wave development after extreme perturbation
These four graphs show the reactant population at various stages of system development, following a *Multi-Extreme* perturbation to a spatial reaction system with non-linear sub-critical interactions ($I = -0.2$). The graphs show the initial system perturbations (top left, $t = 0.0$), the system during the initial decay period (top right, $t = 2.0$), the sudden bounce in populations (bottom left, $t = 12.0$), and the final wave structures as they decay back to equilibrium (bottom right, $t = 25.0$).

implementations of such systems.

3.3.3 Wave Amplitude Decay in Sub-Critical Systems

One recurring attribute of many of the realisations detailed in the previous sections is the decay of wave amplitudes in sub-critical systems. We can make empirical measurements of this decay rate (κ) by recording cell populations at each time unit. We then define κ as the ratio of the population change in unit time, to the wave amplitude before the decrease. We can thus calculate the decay rate as the realisation develops, and how it is affected by the system instability. This allows us to determine the rate of decay of high population locations directly from the system parameters. This ability proves vital for our later work on efficient computer implementations of stochastic system simulations.

Time(t)	Populations		Decay		Decay/Amplitude	
	$X_1(t)$	$Y_1(t)$	ΔX	ΔY	κ_X	κ_Y
13	16.34	16.16				
14	15.07	14.68	1.26	1.48	0.199	0.240
15	13.74	13.42	1.34	1.26	0.264	0.269
16	12.71	12.51	1.02	0.91	0.273	0.266
17	11.98	11.87	0.73	0.64	0.269	0.255
18	11.47	11.43	0.51	0.44	0.256	0.235
19	11.12	11.12	0.35	0.31	0.238	0.218
20	10.88	10.90	0.24	0.22	0.214	0.196
21	10.70	10.73	0.17	0.16	0.193	0.178
22	10.58	10.61	0.13	0.12	0.186	0.164
23	10.48	10.51	0.10	0.10	0.172	0.164
24	10.41	10.43	0.08	0.08	0.167	0.159
25	10.34	10.36	0.07	0.07	0.171	0.163

Table 3.3: Decay rates for a sub-critical non-linear realisation
Unit time changes in X and Y populations from peak wave amplitude to equilibrium, for a sub-critical two-reactant one-dimensional spatial reaction system. The final two columns represent the population *decay rate* — the ratio of unit time population change to the wave amplitude.

As an example of this empirical study, let us look at the rate of decay of wave amplitudes in the sub-critical scenario detailed in the previous section in Figure 3.12. Table 3.3 details population values for the reactants X and Y within cell 0 of the realisation. Measurements of cell populations are recorded for time intervals from the time of

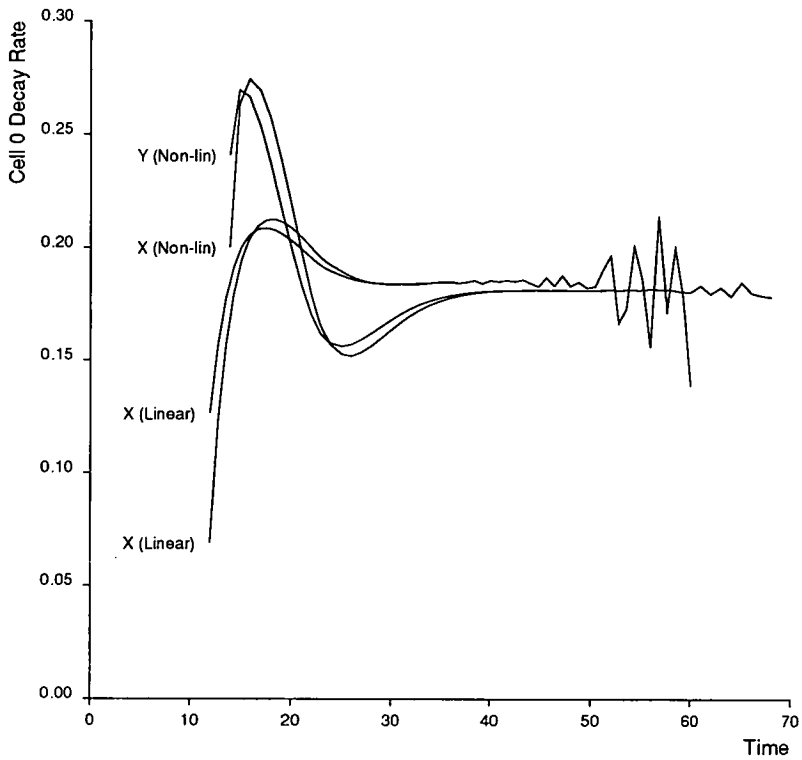

Figure 3.13: Large perturbation linear and non-linear decay rates
Empirical population decay rates for sub-critical ($I = -0.2$) realisations of two-reactant spatial reaction systems with both linear and non-linear interaction functions, and multi-extreme initial conditions. These results show that the linearised system is an accurate approximation to the persistent state of the non-linear system, and from the graph we can identify stable decay rates of $\kappa = 0.181$ for non-linear, and $\kappa = 0.184$ for the linear scenarios.

maximum wave amplitude ($t = 12.0$) until equilibrium is almost reached ($t = 25.0$). The table details the change in population in unit time for X and Y, and the respective decay rates κ_X and κ_Y.

The results from Table 3.3 can be seen in graphical form (which also includes more detailed results) in Figure 3.13. This graph also includes the equivalent decay rates for a realisation with linearised interactions. The initial oscillation of the rates show that, even after the *bounce* phase of the realisation, transient activity is still dominating development. The rather more random variation at larger time values ($t > 40$ in the linear case, and $t > 60$ for non-linear) is caused by the ever-increasing rounding-errors in the numerical calculation of the decay rates, as both the change in cell population and wave amplitude become very small, and comparison of such small amounts is always prone to numerical error. However the graph does allow us to identify the persistent system decay rate before these errors dominate, and this allows us to compare our

realisation results to a deterministic analysis. The rate we obtain (by measurement of the asymptotic values from the graph in Figure 3.13) for the non-linear case is $\kappa_X = \kappa_Y = 0.181$ and for the linear case $\kappa_X = \kappa_Y = 0.184$. If we consider the simple calculation of this rate for the wave amplitude of reactant X — $A_X(t)$, and relate this to an exponential decay function e^d, we obtain

$$A_X(t+1) = (1 - \kappa_X)A_X(t) = A_X e^d ,$$

from which we can calculate the empirical exponential decay factor d. In this way we find that for the non-linear case with *multi-extreme* initial conditions the persistent exponential decay factor is $d = -0.199$, and the equivalent for the linear realisation is $d = -0.203$. Both of these values are remarkably close to the leading eigenvalue of the linearised solution (I); the non-linear result is somewhat closer than the linear. This again confirms that, for the persistent solution at least, the linearised mathematics developed by Turing provides an accurate model of system behaviour.

We have performed similar investigations for realisations with *Multi-Small* initial conditions, using both linear and non-linear interaction functions. The decay rate results for these cases can be studied in Figure 3.14, and here we see an even closer agreement between non-linear systems and their linearised approximation. The observed value of κ for both reactants in both scenarios becomes stable at $\kappa = 0.185$, which equates to $d = -0.2045$, again very close to the instability value used to calculate interaction coefficients for the realisations. These results show that for both linear and non-linear sub-critical systems we can use the system instability parameter as a direct measure of the exponential decay rate for high-reactant populations.

3.3.4 The Importance of Discrete Systems

Throughout our work we have concentrated on spatial systems with discrete locations in which reactants interact, and between which they can migrate or diffuse. This type of system is quite distinct from a diffusion model where the spatial domain is continuous. When modelling such continuous systems by numerical simulation, some degree of spatial discreteness must be introduced to provide a mechanism for solution. However, here we are interested in true discrete systems. Many physical and biological systems

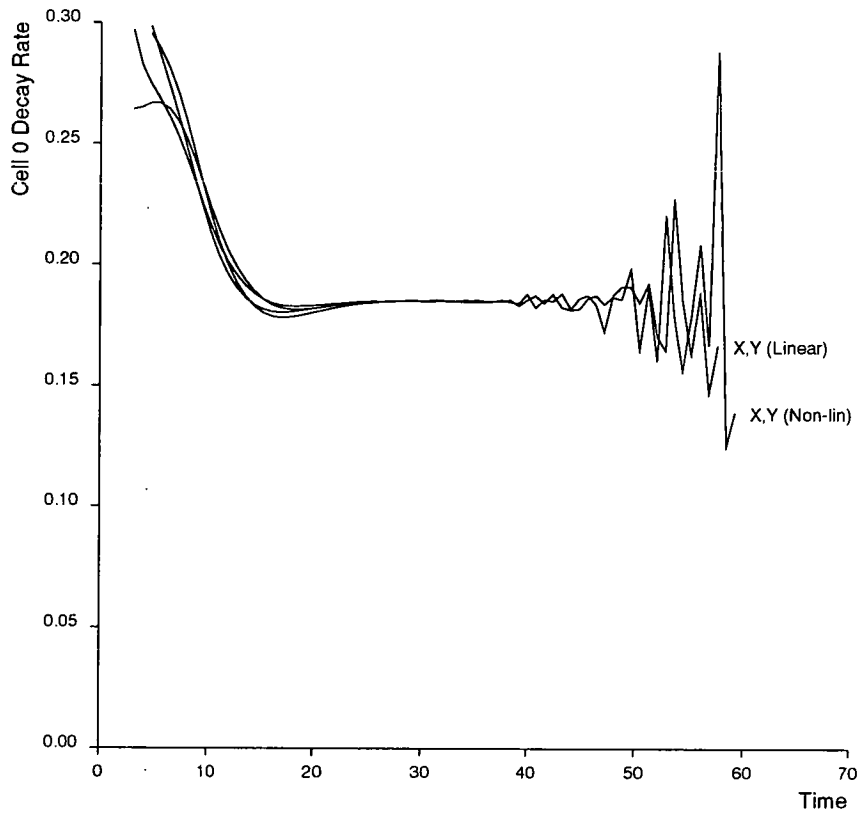

Figure 3.14: Small perturbation linear and non-linear decay rates
Empirical population decay rates for sub-critical realisations of two-reactant spatial reaction systems with both linear and non-linear interaction functions (with instability $I = -0.2$), and multi-small initial conditions. These results show that the linearised system is an accurate approximation to the persistent state of the non-linear system, and from the graph we can identify a single stable decay rate for all scenarios of $\kappa = 0.185$.

can be most accurately modelled in this way (see the pioneering experimental work of Huffaker [1958] and the more recent review of the field by Hengeveld [1989]). For example, we can consider the discrete spatial domain as a string of island habitats (Simberloff [1976]), a collection of animal or plant cells (Wolpert [1969]), or perhaps spatially distributed individuals or colonies in a geographically continuous domain (see Levin & Paine [1974]). Although in many cases the discrete and continuous systems produce approximately identical behaviour, we have found that there are regimes where certain system development features can only be produced with a discrete model. This effect becomes more important later in Chapter 4 when we discuss stochastic systems, but it can still manifest itself in deterministic realisations.

The importance of considering spatial models within the field of population dynamics has been understood for some time now, with Renshaw [1986] providing a review of such *stepping-stone* systems, and the recent work of Mollison, Isham & Grenfell [1994] leaving the reader in no doubt that such systems are vital to epidemiological modelling. We have already discussed one feature that is peculiar to discrete systems (see Section 3.2), namely that when we have an odd number of locations extreme-short wavelength waves cannot fit exactly onto the system. This important distinction extends also to all cases when the number of expected waves does not divide exactly into the number of available locations.

We have also highlighted the displacement of wave crests within discrete systems of high instability, see Figure 3.10. Neither of these effects could be observed with continuous systems, where the requisite number of waves will always be present and correctly located. In realisations where we know that the number of reactant locations does not match that required for the expected spatial-wave structure, we might expect to observe non-standard behaviour. However, where our discrete realisation behaviour is most interesting is when the physical system can support the expected number of reactant waves, but it develops in a quite different manner producing wave structures quite different from those predictable from the system parameters.

Let us consider our standard one-dimensional, two-reactant system with periodic boundary conditions, but now with only ten locations, yet with migration rates set to produce five waves in the system. This scenario is theoretically possible, with neighbouring locations having populations on opposite sides of the equilibrium values. Indeed for

Single perturbation		Multiple perturbation	
I	Waves	I	Waves
0.001	5		
0.007	5	0.010	5
0.008	4^1	0.070	5
0.010	4^1	0.080	4^2
0.020	4^1	0.090	4^2
0.030	4^2	0.097	4^2
0.100	4^2	0.098	3
0.120	3	0.100	3
0.200	3	0.400	3
0.400	3	0.500	-
0.430	-		

Table 3.4: Wave shapes on a ten-cell ring with various instabilities
This table details the number of waves produced for a 10-location non-linear spatial reaction system in a super-critical state ($I > 0$). We list the number of waves in the persistent system state for each instability value. Figure 3.15 can be used as a reference to identify these patterns; note the differences highlighted by the super-scripts for the two different four-wave solutions.

the cases with linear interaction functions five waves are always produced, and for the non-linear case at criticality and below (provided initial perturbations do not destroy the system before it reaches a persistent morphological state) we also obtain the expected five-wave format. However, when we have $I > 0$ and our standard non-linear interaction functions, we can observe some interesting wave patterns that show the unexpected ways in which discrete systems can develop.

Table 3.4 details the number of "waves" that we observe for our ten-location super-critical realisations. These structures are often not regular and Figure 3.15 shows the form of some of the observed patterns. As the instability in our realisation increases we see that the system contains persistent reactant distributions that bear decreasing resemblance to the solutions we would expect from our theoretical results in Chapter 2.

For all the realisations detailed in Table 3.4 the perturbation used is a single X-reactant removed from the initial populations in either a single, or five equally spaced cells. We see that once the instability is over a certain value (just over $I = 0.4$) it becomes too large to produce any persistent solution other than an empty system, with all cell populations reducing to zero. Just before this occurs, the system can enter violent oscillations.

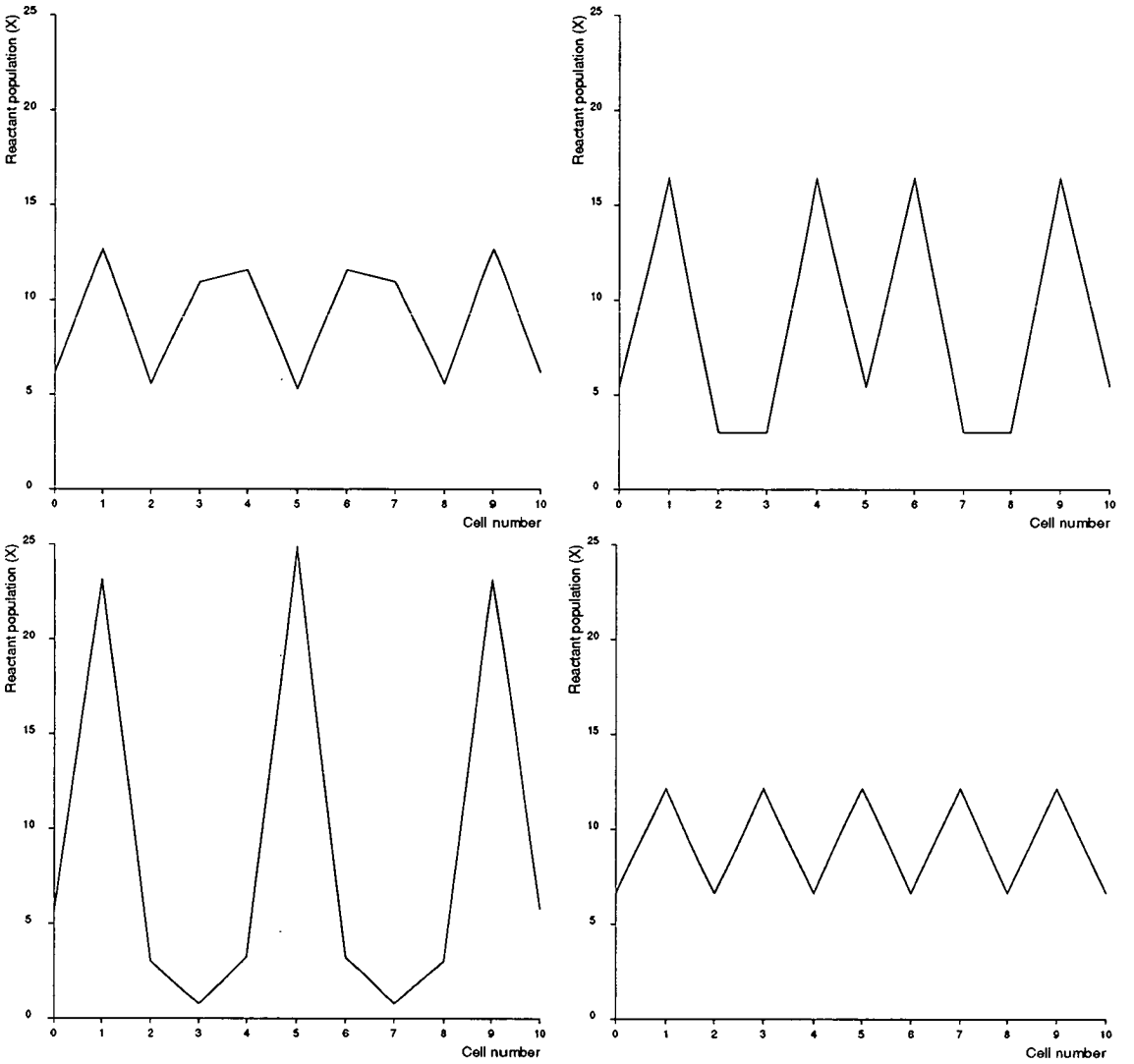

Figure 3.15: Possible wave-shapes on a ten-cell Turing ring

Four graphs of the irregular wave structures produced by super-critical realisations of two-reactant spatial reaction systems with non-linear interaction functions on a ring of ten cells. The top left graph shows the first four-wave variant (denoted 4^1 in Table 3.4), the top right graph shows the second four-wave variant (4^2). On the bottom left we have the three-wave distribution, and finally at bottom right we have the standard five-wave case for comparison.

We observe this for the single cell perturbation case with $I = 0.4$. The final steady-state now contains just three waves (see Figure 3.15, bottom left graph), but before stability is reached the wave structures undergo rapid oscillations in amplitude. Any further increase in instability causes these oscillations to be severe enough to destroy all reactant populations.

Further experiments, similar to the above but investigating a variation in the number of cells rather than in instability, also produce interesting and informative results. Our spatial reaction systems now develop population wave patterns that make direct use of the discrete nature of the ring system — by adjusting the wavelength and periodicity of the final stable state to fit the available discrete locations. The diffusion-driven instability will always push the system to a wave-like state, although the resulting morphological structure may be far removed from that expected from the system parameters. Such behaviour cannot be demonstrated for continuous systems in which specific migration rates can always produce the specified number of wave crests, since the required wavelength is not restricted by available locations.

Number of cells	Waves Produced
7	2
8	3
9	3
10	4^1
11	4^1
12	4^1
13	4^1
14	5
15	5

Table 3.5: Wave numbers produced with small ring sizes

Using a super-critical two-reactant spatial reaction system with instability $I = 0.1$, and *Single-Small* initial conditions, this table records the number of wave crests that appear for small ring systems with a range of sizes. In all the realisations migration parameters were set to produce five waves according to the linearised approximations, although the actual interactions used are of our general non-linear type.

The results obtained from this variation of ring size are detailed in Table 3.5. Again Figure 3.15 can be used as a guide to the general structure of the three- and four-wave cases. We do, however, need to consider one new wave-shape, that with just two waves, and this has been reproduced for the seven-cell system in Figure 3.16.

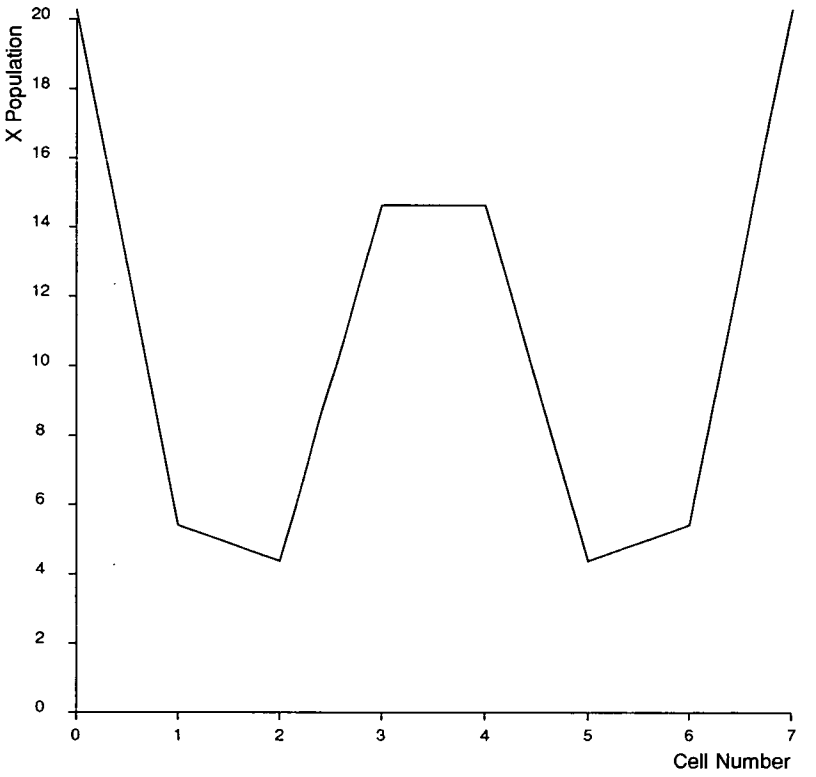

Figure 3.16: A two-crest wave-shape on a seven-cell ring

This graph shows the wave structure produced by a super-critical ($I = 0.1$) realisations of two-reactant spatial reaction systems with non-linear interaction functions. Migration rates are set to produce five waves around a ring with only seven locations, and we observe that the resulting state of the system contains just two complete waves.

3.3.5 Wave Structure Stability in Systems at Criticality

In covering both sub- and super-critical realisations of two-reactant spatial reaction systems in one dimension, we have given many details of wave pattern behaviour, and how it is governed by the leading eigenvalue of the solution to the system equations. We have, often in passing, mentioned that the critical system (where $I = 0.0$, and thus the leading eigenvalue has zero magnitude) does not fit into any of the behaviour patterns so far discussed. We have found that following some initial perturbation to an equilibrium system at criticality, wave structures will appear around the ring of cells, and will (over very long time periods) settle to produce a stable state. This final state will usually contain the expected number of population waves according to the migration rates chosen (as in super-critical systems), but the amplitude of the structures, and the time taken to reach stability, are dependent solely upon the initial conditions of the realisation.

It is important to define what is meant by *stability* in these discussions. In an ideal realisation we would expect that, at the point of stability being reached, reactant populations in all cells undergo no change within a given time period. Unfortunately, when working with computer realisations, there is always some numerical error involved with using real numbers, and each machine has a particular level of accuracy inherent in its hardware. In all our realisations we have used "double precision" arithmetic, this provides a numerical accuracy of $\sim 10^{-16}$ (compliant to the IEEE standard) on all our workstation or supercomputer platforms. These rounding errors can occur throughout a realisation, and although they can be kept small enough so as not to affect the accuracy of the result, they can prevent us from guaranteeing an accurate measure of real stability within our systems. We therefore choose to identify stability as the first point in the realisation that all cell populations change by less than some specified *stability threshold*, λ_s. Provided that we choose λ_s to lie above the known numerical accuracy of our realisation, we can identify the point at which the realisation has reached our given measure of stability. For the results detailed in this chapter, we have chosen this threshold to be $\lambda_s = 10^{-6}$.

Table 3.6 details our experimental results for the stable state wave amplitudes of both reactants (A_X, A_Y) and times to reach stability (t_s), for a variety of initial conditions. In all these realisations we used the *Single-Small* conditions detailed in Section 3.3, but with the single positive perturbation of each reactant being of varying size — from 10%

Perturbation	Linear Interaction			Non-linear Interaction		
%	t_s	A_X	A_Y	t_s	A_X	A_Y
10	957.2	0.040	0.040	956.9	0.039895	0.039896
20	1005.0	0.080	0.080	1004.2	0.079482	0.079483
30	1032.9	0.120	0.120	1031.5	0.118611	0.118613
40	1052.8	0.160	0.160	1050.5	0.157133	0.157140
50	1068.2	0.200	0.200	1065.0	0.194909	0.194921
80	1100.7	0.320	0.320	1093.7	0.302474	0.302518
100	1116.1	0.400	0.400	1106.1	0.368325	0.368405
150	1144.2	0.600	0.600	1126.0	0.508878	0.509091

Table 3.6: Critical wave amplitudes and stability timings

This table compares the times to reach stability, and the stable wave amplitudes for each reactant, for spatial reaction systems at criticality ($I = 0.0$). Results are given for realisations using both linear and non-linear interaction functions, for our standard 50-cell, five-wave scenarios.

to 150% of the population of the equilibrium cells.

The results detailed in Table 3.6 identify a number of important effects. First we can see that there is good agreement between the amplitudes and timings for the linear and non-linear realisations, although the discrepancy grows as we increase the strength of initial perturbations. This discrepancy leads the non-linear system to reach stability faster than the linear system, by an amount that increases with perturbation size. However, non-linear stable wave amplitudes are reduced relative to the linear case. We can also see that there is a growing discrepancy between the X and Y populations at stability for the non-linear scenario, whereas for linear interactions reactant populations are always identical. The linear amplitudes are also directly proportional to the perturbation size, agreeing to the full accuracy of our double-precision realisation. We feel that this result provides us with direct evidence that the final stable state of these critical realisations (i.e. when $I = 0.0$) is crucially dependent upon the initial conditions for the system.

We can also use the above stability measure technique to study the time taken to reach stability (t_s) in non-linear systems that are either sub- or super-critical. A similar study of systems with linear interaction functions fails when we consider the super-critical case, as this can never reach true stability. Figure 3.17 shows the variation of t_s with the system instability I between small sub-critical values ($I \simeq -0.05$) and super-critical values high enough to cause a breakdown in any persistent state ($I > 0.5$). This graph

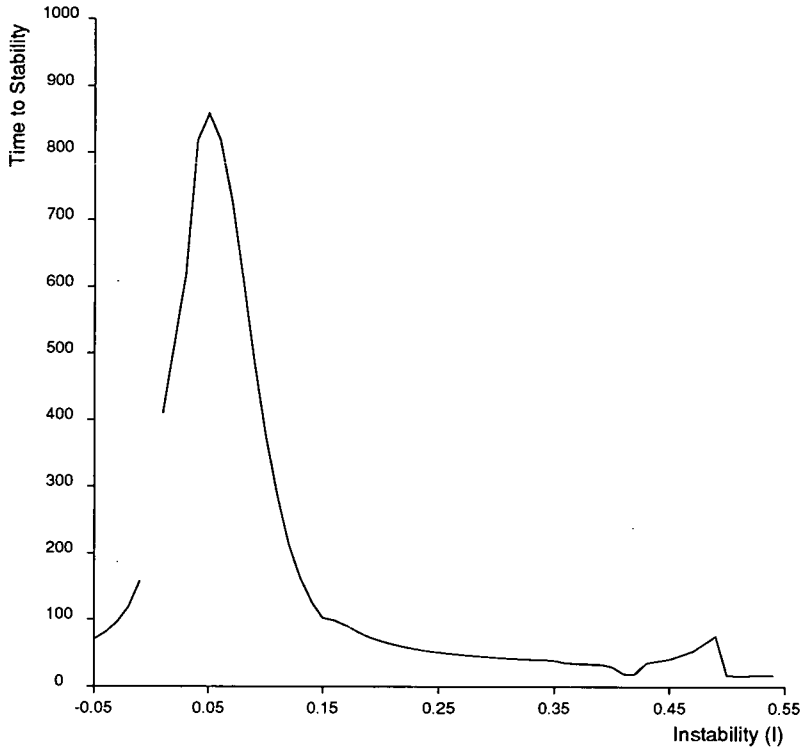

Figure 3.17: Time taken to reach non-linear stability

The variation of times to reach stability (t_s) with interaction instability I for a two-reactant one-dimensional spatial reaction system with non-linear interaction functions. The stability threshold is taken as $\lambda_s = 10^{-4}$ and the system contained 50 locations with *single-small* initial conditions. The discontinuity in the graph lies around criticality ($I = 0.0$), which is a special case as discussed earlier in this section. The apparent *breakdown* at high instability is also discussed below.

shows a distinct *resonance*-like area where realisations take substantial times to settle to stability. The main area of strong resonance is centred around small super-critical instability values $I \simeq 0.05$, for our particular standard parameter set. The graph in Figure 3.17 also shows some unusual activity for instability values above $I = 0.4$. Initially as I rises over this value we appear to be approaching another "resonance" and thus the time to stability increases again. However, we then find that our system becomes unstable over $I = 0.5$ and thus the graph "breaks down" in this region.

3.4 Realisations of Travelling Reactant Waves

In the earlier sections of this chapter we have studied, in some depth, the presence of stationary and oscillatory wave patterns. We have concentrated upon the asymptotic results from our generic system with two-reactants in a one-dimensional system with discrete locations. In Section 2.4.1 we provided analytic solutions for the transient systems with two or three reactants and non-linear interactions. Given the assumption that the solution could be approximated (using Laplace Transform techniques) by a travelling waveform moving through the ring locations with a constant velocity (v), these investigations produced no acceptable solution. This analytic study was motivated by the realisations detailed in this section, as they show the clear presence of travelling transient waves. Hopefully, a successful non-linear analysis will result from future work.

For an empirical study of transient travelling waves the realisation program discussed in Section 3.1 has been adapted to allow the progress of population waves to be monitored. This proved to be an awkward task due to the difficulty in specifying a population *wavefront* within a computer program. Although it is relatively easy to produce realisations that provide an obvious travelling wave to the human eye (see Figure 3.18 for example), specifying such details as computer code requires substantial additions to the original programs. In general all the wavefront-identification methods developed in this work depend upon identifying a specific change in the population of a cell, and then determining whether this constituted the arrival of a wavefront. Later we detail the various techniques that have been implemented, along with the results that we have produced. This follows a brief discussion of numerical accuracy considerations for ring

and infinite-line systems.

For a truly deterministic spatial system, as soon as a perturbation occurs at any cell, the effect is felt immediately in all other cells in the ring. However, because of the discrete nature of computer simulations, this effect can only travel around the ring through one cell for each iteration. Therefore, for the case of a ring of 50 cells, the effect of the perturbation in one cell will only be felt in the cell opposite after 25 iterations, rather than immediately. Thus, should the iteration step-size be large this specific number of iterations may constitute a significant time when compared to the time-scale over which any travelling wave exists. Therefore in all the realisation results detailed below we identify the rate of fastest possible progress through the spatial domain according to the iteration step-size in use. We can therefore confirm that any travelling wave effects observed are in fact genuine, and not solely an artifact of the particular realisation parameters.

As was detailed in Section 2.4.2 there are two possible scenarios of wave motion in our spatial reaction systems. The first of these involves motion away from a populated equilibrium state following a single perturbation, and the second covers the spread of a reactant through an empty ring. These two cases have been termed *E-waves* and *O-waves* respectively, and we now consider each in detail.

3.4.1 Non-Linear Movement from Equilibrium — E-waves

We have studied super-critical non-linear realisations using a ring of 200 cells with equilibrium populations $X_i = X^*$ and $Y_i = Y^*$ for $i = 1, N$, with a single cell (j) perturbed so that $X_j = X^* - 1$ and $Y_j = Y^* - 1$. This configuration produces a travelling wave around the ring, emanating from the initially perturbed cell. The wavelength of the permanent wave left behind the wavefront is defined by the choice of migration rates. Similarly, the wave amplitude is determined by the instability of our system. However, let us concentrate on the transient phase of system development, as the wavefront moves through the ring as though it were an infinite line system. The realisation software is written to identify the time of any population change at the ring location opposite the initial perturbation, and at this time the realisations are halted. At this point the transient infinite-line behaviour merges with the persistent wave solution

to give the periodic boundary condition results detailed earlier in this chapter. All the results in this section are therefore equivalent to having been performed on an infinite line, rather than a ring of cells.

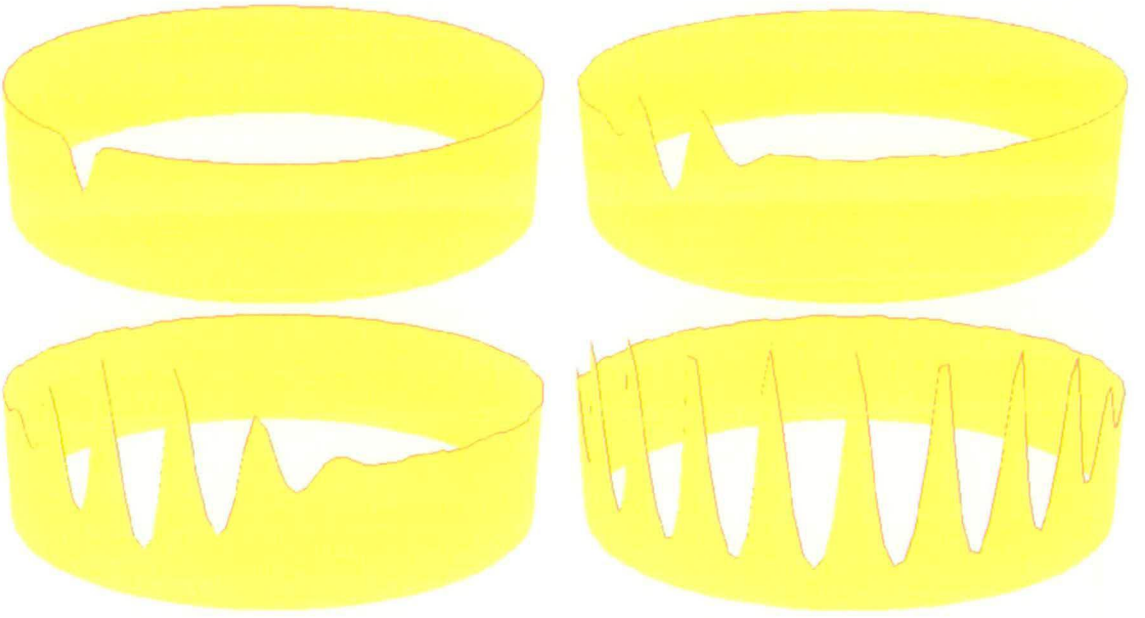

Figure 3.18: Travelling wave evolution in a one-dimensional system
Four snap-shots of the travelling wave created by perturbing a single cell in a super-critical non-linear spatial reaction system. The realisation uses 200 locations, migration rates calculated to produce a wavelength of 10 cells, and a system instability of $I = 0.2$.

The first naïve attempt to record the times of wave-front arrival at each cell involves recording the first time that the population of each cell moved away from initial equilibrium by more than a given percentage of the equilibrium value; we define this percentage as the wave threshold, λ_w. The graph in Figure 3.19 shows the timings achieved using a wave threshold of $\lambda_w = 10\%$ for a range of instability values. The rather uneven curves produced by this method appeared in similar simulations with both larger and smaller λ_w values. The shape produced is due to positions on the ring furthest from the eventual nodes of the wave shape moving away from equilibrium much faster than those closer to the wave nodes, which may in fact never move much away from equilibrium. We can see that for increasing values of I these spikes are less apparent, and the line gradient (which corresponds to the reciprocal of the speed of wave progression) is easier to observe. At higher I-values the wave patterns produced are increasingly severe, and therefore the movement away from equilibrium may be more noticeable at all points in the ring.

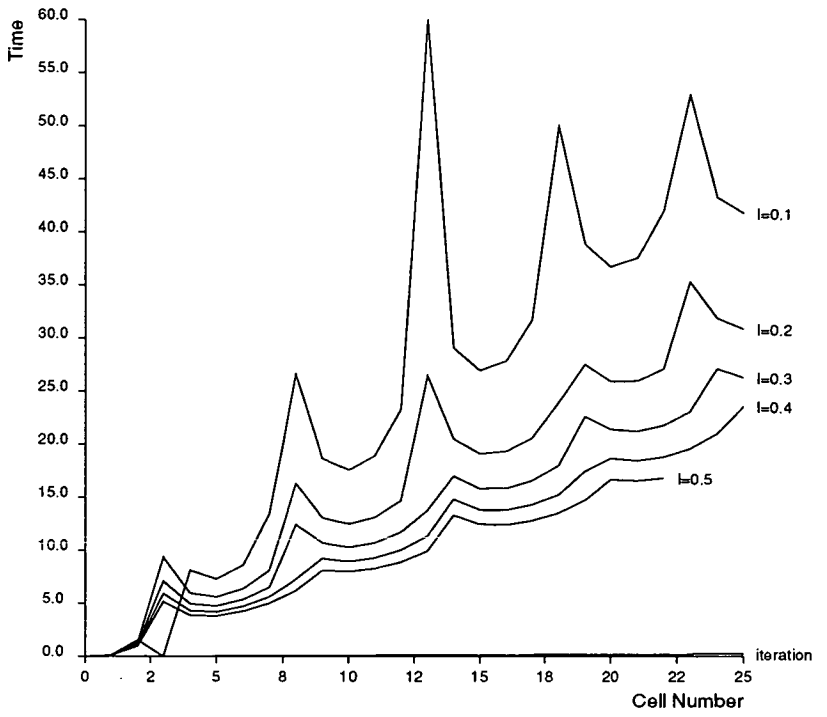

Figure 3.19: Raw arrival times for non-linear transient waves

For a non-linear super-critical spatial reaction system, we show the first time that each cell's X-population moved more than 10% away from X^*. The line corresponding to the arrival of the iteration-size effect is also shown, and it can be seen to lie substantially below the main identification lines.

Filtered non-linear movement from equilibrium

In an attempt to remove the location-dependent fluctuations from the graphs of wave arrival times, we can introduce a filtering function that weights each arrival time from the graphs in Figure 3.19 according to its position on the final wave state, assuming a permanent wavelength of ten cells. This filtering provides a new arrival time $t_f(r)$ for each cell r, according to the relation;

$$t_f(r) = \frac{1}{20}\left(t(r-5) + t(r+5) + 2(\sum_{j=-4}^{j=4} t(r+j))\right) \quad .$$

The results produced by this method give very regular wave arrival times, as can be seen from Figure 3.20. We are now able to use these results to extract specific empirical velocities for each instability value. Taking the reciprocal gradient of each line provides a measure of the number of sites the wavefront moves through in unit time (spt). We make these measurements using a standard regression gradient calculation facility within

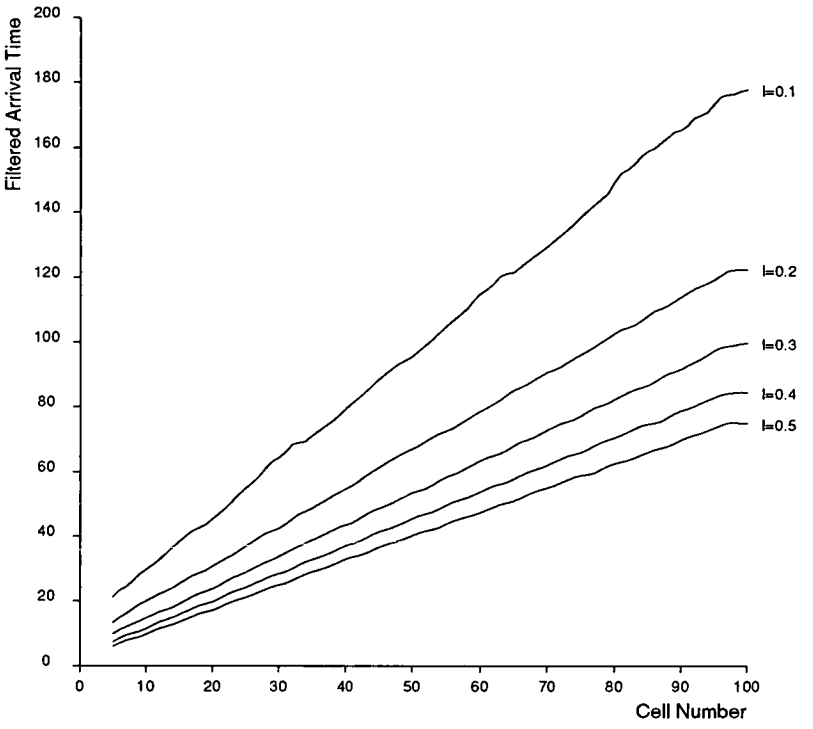

Figure 3.20: Filtered non-linear E-wave arrival times

Filtered non-linear deterministic wave arrival times for a range of super-critical instabilities in a 200-cell system. Wave arrival is signalled by the first $\lambda_w = 10\%$ change in equilibrium populations.

the *Xmgr* graph-plotting software package, and have obtained velocity measurements ranging from $0.59spt$ with $I = 0.1$ to $1.34spt$ when $I = 0.5$. The full set of results is detailed later in Table 3.7. Following further experimentation we are able to provide the graph in Figure 3.21 that shows the variation of empirical non-linear E-wave velocity over a wide range of system instability.

3.4.2 Linear Movement from Equilibrium

Let us now consider the realisation of transient travelling waves in the linear system. As in the non-linear case above, the ring is initially at equilibrium, and a single cell is then perturbed. We again achieve wave-like patterns around the ring, though unlike the stable structure of varying positive populations that results from the non-linear system, the linear case gives us exponentially growing with amplitude waves. Rescaling the populations for display purposes produces graphical output such as in Figure 3.6 in Section 3.3.

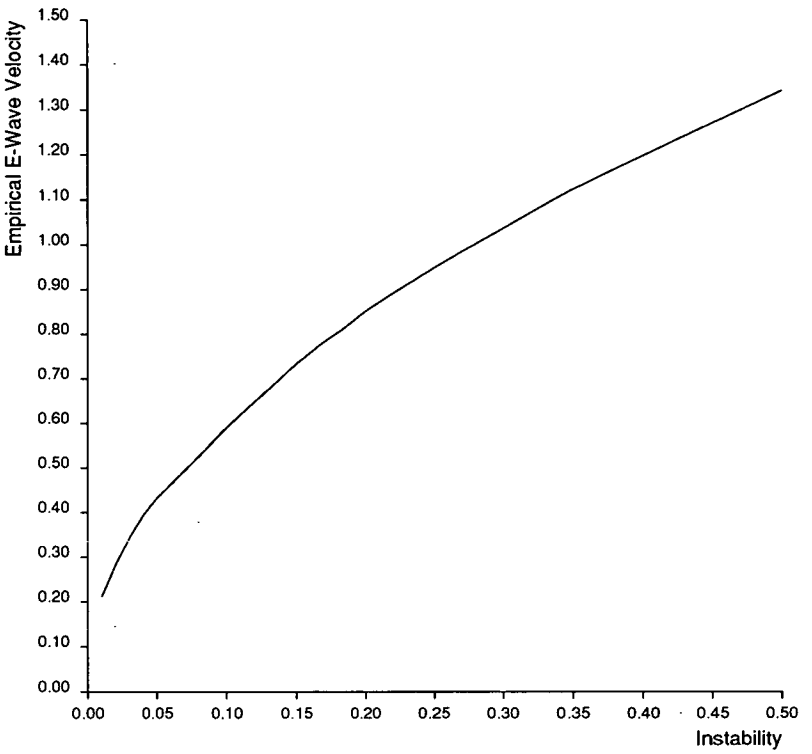

Figure 3.21: Variation of non-linear E-wave velocity with instability
Empirical E-wave velocities are taken from the reciprocal gradients of filtered wave arrival times. This graph shows the variation of wave propagation speeds (measured in sites per unit time (*spt*)) with the interaction instability within the super-critical systems.

We can again measure the velocity of progression of the travelling wave during the transient phase of system development. As in the previous non-linear study this was attempted by identifying the times for each cell at which the population of one species (X, say) first moved away from equilibrium by a certain percentage (λ_w) of the equilibrium population. Figure 3.22 shows the results produced by this approach for a single instability value ($I = 0.1$) using a range of recognition thresholds (λ_w). As in the previous section the lines produced are highly distorted due to certain cells populations being more mobile than others. However, this distortion is more regular than for the non-linear case, and empirical gradients (and hence wave velocities) are easily identified without requiring filtering. It is interesting to note that even for very small recognition thresholds (e.g. $\lambda_w = 10^{-7}$) our realisation identifies the wave arrival at times substantially after the arrival of the numerical iteration effect.

This more regular variation in the curves of Figure 3.22 allows us to determine accurate gradients and these are detailed in Table 3.7. It is interesting to note from Figure 3.22

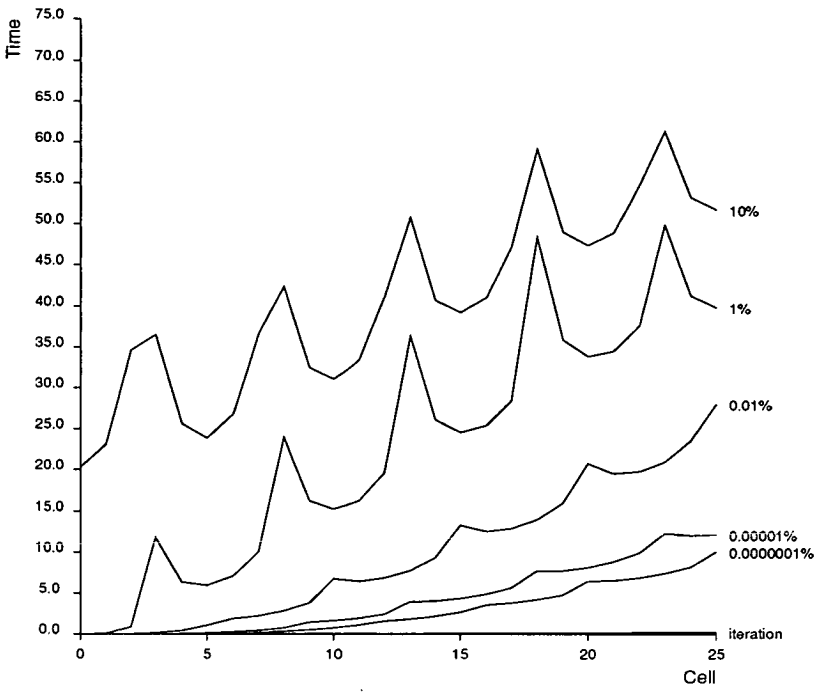

Figure 3.22: Linear E-wave arrival times

Wave arrival times for a linear system moving away from equilibrium. The system instability, $I = 0.1$, and wave arrival times are recorded for a variety of sensitivities of movement (λ_w). The lowest line on the graph, labelled *iteration* corresponds to the fastest possible arrival time due to numerical iteration effects.

the variation in waveform velocity with the recognition threshold λ_w. The graph details approximate velocities ranging from $0.55spt$ at 1% movement to $1.67spt$ at $\lambda_w = 10^{-7}\%$. This increase in speed at higher sensitivities may be due to the recognition of transient behaviour ahead of the main wavefront, and this conjecture is supported by the steadily-increasing gradient of the high sensitivity scenarios. This effect is not the same as the iteration size effect (which can be seen to travel very much faster than all measured velocities $\sim 100spt$), although it may be directly related to it. If we consider the very small changes that can (and will) propagate through the system at speeds equal to the iteration effect, these could induce (secondary) perturbations into the system as it is in unstable equilibrium (I being positive). These perturbations would grow in the system and hence tend to move populations away from the equilibrium value. So when we study sensitivities as small as $\lambda_w = 10^{-7}\%$ we may well be observing the growth of waves produced not only by the original perturbation of one cell, but by the small secondary perturbations caused by the iteration size effect moving around the ring. This effect can be thought of as a *cascade* emanating from the single initial perturbation. For this reason is makes more sense to concern ourselves with the wavefront velocities

corresponding to larger sensitivities of recognition. The results for the empirical studies in this section are fully detailed later in Table 3.7, where again gradients (and hence velocities) have been calculated with the Xmgr facility.

3.4.3 Travelling Waves Through an Empty Ring — O-waves

Let us now look at the other type of system discussed in Section 2.4 — O-waves. In this case we start with a ring devoid of both species and then introduce reactants of both types into one, or a group of cells. In the non-linear realisations detailed earlier in Section 3.3.2, we saw that with *Single-Extreme* initial conditions the sub-critical scenario could produce waves of each reactant moving out from a single point to populate the whole system. The spread of this extreme perturbation can be seen as a simple population wavefront travelling through the system, leaving equilibrium populations in its wake (see Figure 3.23). We have therefore performed empirical measurements of this wavefront velocity, using initial conditions of a group of perturbed cells as this provides a more stable development of the system. The critical and super-critical equivalents of both linear and non-linear systems, as well as the linear sub-critical scenario, do not support the above O-wave behaviour, since all initial perturbations simply decay back to the empty ring state.

Our experiments have produced results for these non-linear O-wave scenarios that provide surprisingly well-defined arrival time gradients, since the wavefront progression does not suffer from the super-position of a stationary wave structure on to the final stable state. Figure 3.24 shows the wave arrival times for our O-wave realisations in non-linear scenarios. The actual values of wave propagation speed determined from these results are detailed in Table 3.7. Gradients have been measured for the latter sections of the graphs in Figure 3.24, thus ignoring the early transient motion affecting cells zero to ten.

Table 3.7 therefore provides us with wave velocity values for both super-critical E-waves and sub-critical O-waves, for both linear and non-linear interaction functions (other than linear O-waves which are never produced). We observe that the velocity of propagation increases with the system instability for all super-critical waves. In the sub-critical scenario we see that wave velocity also increases as the system parameters move further

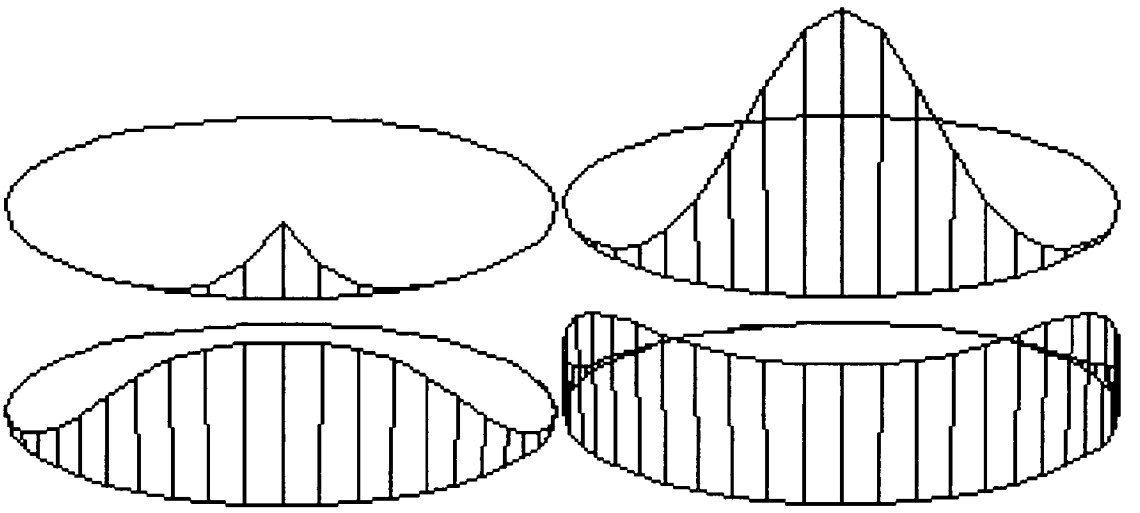

Figure 3.23: Travelling O-waves in a one-dimensional system
Four snap-shots of a travelling O-wave created by perturbing a single cell in an empty sub-critical non-linear spatial reaction system. The realisation uses 50 locations with migration rates calculated to produce a wavelength of 10 cells, and a system instability of $I = -0.15$. The diagrams correspond to the system at times $t = 0.5$ (top left), $t = 16.0$ (top right), $t = 30.0$ (bottom left) and $t = 60.0$ (bottom right).

from the criticality conditions (and therefore I becomes more negative). The table also shows a reasonable correspondence between linear and non-linear velocities, with the non-linear versions producing slightly higher wave velocities.

3.5 Realisations of Three-Reactant Systems

In his pioneering paper Turing [1952] predicts that a ring system with three reactants can produce travelling reactant wave patterns, given the correct system parameters. Our simulations of such systems show that his suggested parameters (see Table 3.8) produce standing waves of oscillating amplitudes, rather than true travelling waves. However the simulations do result in some very interesting behaviour dependent upon the system criticality.

In general, the realisations of such three-reactant systems result in two reactants (X and Y) oscillating almost in phase, and the third (W) oscillating π out of phase with them both. These oscillations are highly regular, and similar to the oscillating two-reactant linear wave shown in Figure 3.2. However, by using non-linear interactions we can often keep the wave amplitudes bounded for some of the possible realisation scenarios.

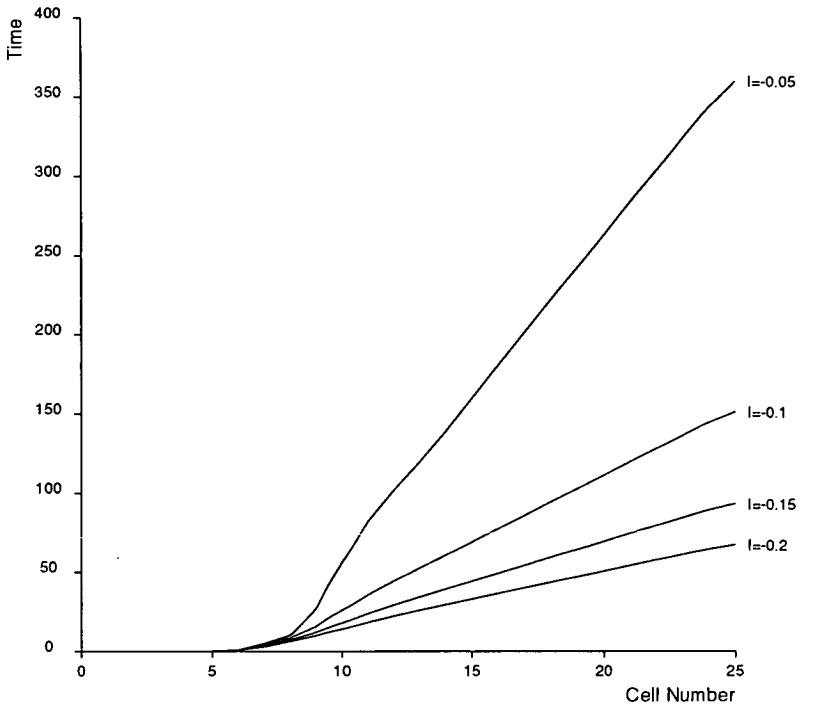

Figure 3.24: O-wave timings for sub-critical systems

Wave arrival times for non-linear O-waves advancing through an otherwise empty ring. These waves are produced following a block of five cells being perturbed, and hence recognition times begin from cell number five rather than zero. The recognition sensitivity is $\lambda_w = 10\%$. Note that the instability values are negative.

3.5.1 Non-Linear Three-Reactant Systems at Criticality

The critical parameter set suggested by Turing produces standing waves using the non-linear versions of the interaction functions, with a conversion from the parameter set in Table 3.8 following similar lines to that detailed in Section 3.1.3. The system initially appears stable, and for substantial times (up to 200 time units) the reactant populations oscillate with fairly regular amplitudes. We thus obtain three standing waves with identical periods. Figure 3.25 shows the population variation in cell 0 — at a mid-point between two standing wave nodes, and thus where the oscillation amplitude is at a maximum. For cells lying on wave nodes there is almost no movement in the reactant populations.

However, when we consider long realisation times (i.e. over 200 time units), the critical system moves away from these stable oscillations. We eventually observe an exponential explosion in the reactant populations — the type of behaviour we would expect from a super-critical system. This sudden swing to instability can be seen clearly in Figure 3.25,

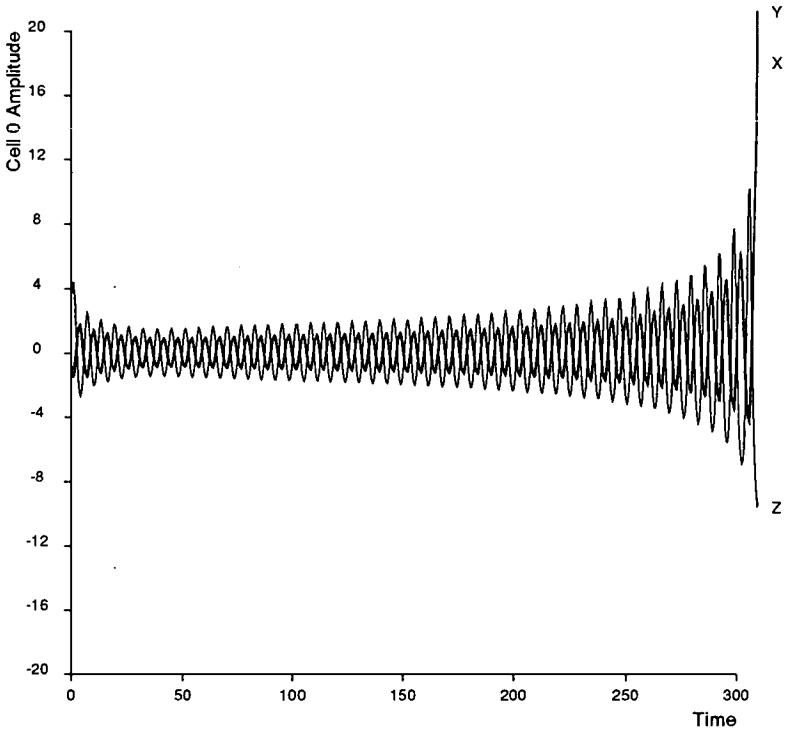

Figure 3.25: Critical standing waves of three reactants

This graphs shows the amplitudes of oscillation of the population of cell 0 in our standard 50 cell one-dimensional spatial reaction system with non-linear interactions at criticality. Note that for substantial times these oscillations appear constant, until after approximately 150 time units they begin to grow, leading to an explosion around $t = 300$.

		λ_w	I	Velocity	Error
Non-linear	E-waves	10	0.1	0.5901	0.002
		10	0.2	0.8502	0.002
		10	0.3	1.0362	0.001
		10	0.4	1.1991	0.002
		10	0.5	1.3435	0.003
Non-linear	O-waves	10	-0.05	0.050	0.081
		10	-0.1	0.120	0.029
		10	-0.15	0.200	0.019
		10	-0.2	0.284	0.017
Linear	E-waves	10	0.1	0.563	0.120
		10	0.2	0.797	0.069
		10	0.3	1.005	0.046
		10	0.4	1.170	0.033
		10	0.5	1.306	0.031

Table 3.7: Wave progression velocities

Empirical values for the velocity of wave progression have been determined from our realisation results, using the standard regression facility within the *Xmgr* software package. We have considered both non-linear and linearised system interactions, and give speeds for the two types of observed travelling waves: E-waves in super-critical scenarios where reactant populations move away from a populated equilibrium state; and O-waves in sub-critical systems where a wavefront of reactant spreads through empty cells. Error values detailed are standard root-mean-square errors as calculated by the Xmgr package.

and is almost certainly due to a lack of accuracy in the numerical simulation resulting in an effectively non-zero instability value being created by the rounding errors introduced during the very large number of iterations.

3.5.2 Non-Critical Three-Reactant Systems

For super-critical systems we generally obtain very complex explosive behaviour. For example, in the non-linear super-critical case with $I = 0.1$ we initially see a double wave structure, with twice as many waves of each reactant as expected from the system parameters. This system then undergoes a rapid explosion in oscillation amplitudes (see Figure 3.26). This graph clearly shows the standing wave oscillations of all three reactant populations. As in the critical case, the X and Y populations oscillate almost in phase with each other, and the W population oscillates approximately π out of phase

Reactant Type	Interaction with X	Y	W
X	I-10/3	3	-1
Y	-2	I+7/3	0
W	3	-4	I
Migration Rate	2/3	1/3	0

Table 3.8: Coefficient values for three-reactant waves

The parameters suggested by Turing [1952] for the production of travelling waves of reactant populations. Our realisations, detailed in this section, show that standing wave patterns are produced using this set of coefficients. These coefficients produce a critical system if we take $I = 0$.

with them both.

In the linearised super-critical system we also get a rise in oscillation amplitude, although the increase is linear rather than exponential. If we select an arbitrary amplitude as the limit for explosion (say 50) we can plot the time taken for the system to *explode* as it varies with instability — see Figure 3.27. We see a relationship whereby explosions occur faster as instability increases.

In sub-critical systems with three reactants we get much more controlled behaviour (both linear and non-linear) as the oscillation amplitude decays towards zero and the system return to equilibrium. This type of inherently stable behaviour is what we would expect from such systems, and it can be seen in Figure 3.28.

In summary, we see that the three-reactant scenario does not produce travelling waves, but produces standing waves that fit exactly into our concept of criticality. The period of oscillation of these standing waves is found to be invariant with the instability, although for high instability super-critical scenarios we have little data to analyse before the explosive behaviour dominates. Testing over a range of instabilities produces the results detailed in Table 3.9 which show consistency in period size between different instability values, but a distinct lengthening of the oscillation period as the system moves away from regular amplitude oscillations. This therefore occurs earlier in those experiments with higher instability values.

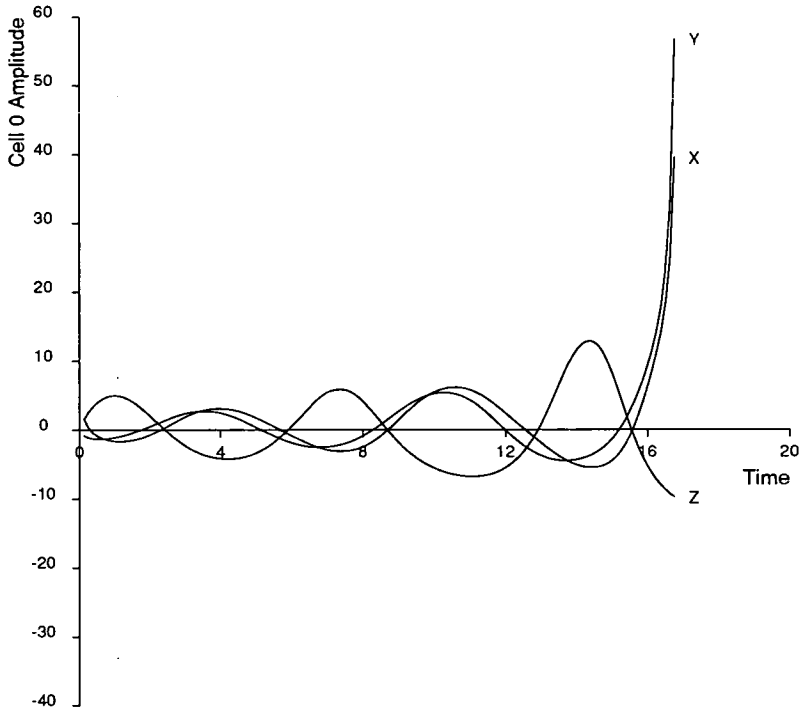

Figure 3.26: Super-critical standing waves of three reactants

This graph shows the amplitudes of oscillation of the population of cell 0 in the three-reactant variant of our standard 50-cell one-dimensional spatial reaction system with interaction coefficients chosen to produce super-critical behaviour ($I = 0.1$). Note the rapid explosion of the oscillation amplitudes.

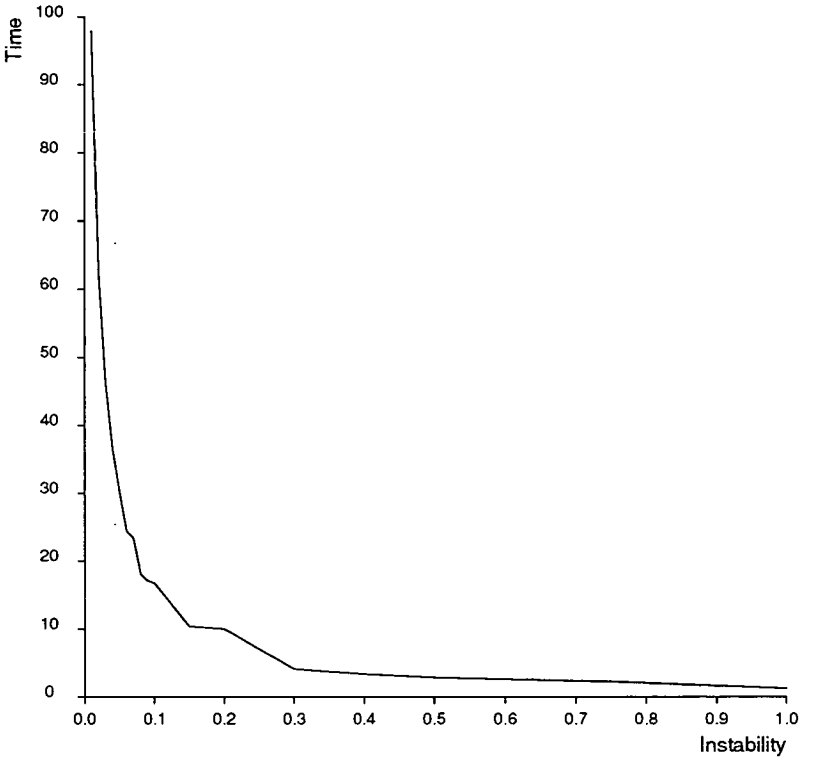

Figure 3.27: The time to explosion of super-critical standing waves
Having chosen an *explosion* to have occurred whenever the amplitude of oscillation has exceeded 50, we can plot the time to explosion for a range of instability values in the super-critical domain. The results are for our standard 50-cell one-dimensional spatial reaction system.

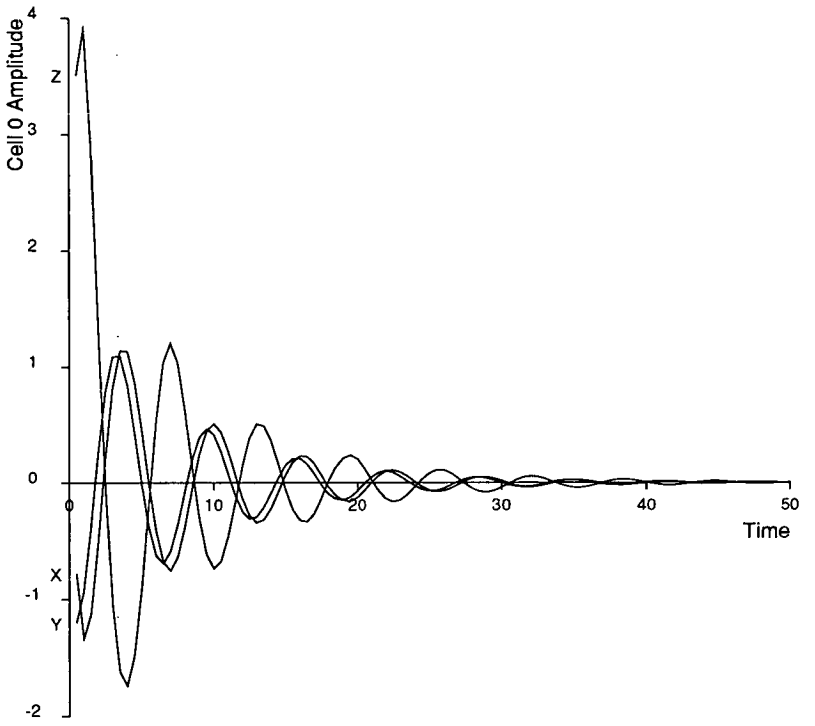

Figure 3.28: Sub-critical standing waves of three reactants

This graphs shows the amplitudes of oscillation of the population of cell 0 in our standard 50-cell one-dimensional spatial reaction system with interaction coefficients chosen to produce sub-critical behaviour (i.e. $I = -0.1$). Note the decay of the oscillation amplitudes.

Instability	Empirical Period						
0.001	6.45	6.40	6.45	6.40	6.35	6.40	6.40
0.01	6.55	6.50	6.55	6.50	6.50	6.60	6.65
0.02	–	6.55	6.65	6.70	6.95	7.05	–
0.1	–	–	6.60	6.90	7.10	–	–

Table 3.9: Empirical periods of three-reactant standing waves
For a range of instability values, our standard 50-cell spatial reaction system on a ring with three reactants shows consistent periods of standing wave oscillation. Only as the amplitudes of oscillation grow do the periods change substantially from 6.5 time units — the more violent oscillations producing more rapid lengthening of the period.

3.6 Two-Dimensional Realisations

In this section we report briefly on the empirical results of our realisations of two-dimensional spatial reaction systems. These calculations typically cover very large spatial domains, and have therefore been performed on powerful parallel computers in order that a representative sample of experiments could be completed in a realistic time-scale. The actual parallel computer implementations of these simulations will be described in full detail later in Chapters 6 and 7. In this section we concentrate on a small sample of our results in order to give a *flavour* for the types of study that are possible for these systems.

We feel that two-dimensional studies such as those we detail later will grow rapidly in importance in the scientific community. This will be made possible by computers becoming available that can service the massive calculation requirements for realisations and simulations of such systems. In recent years the chemical studies of Castets *et al* [1990] and Ouyang & Swinney [1991], along with the biological research of Hunding [1990] and Nagorcka & Mooney [1992], have all shown that two-dimensional systems can produce effects close to those observed in natural systems. Such results help to confirm the importance and relevance of computational analysis, and thus encourage wider research and use of simulation techniques.

The parallel computers in use for our two-dimensional work all produce very high quality interactive graphical output. This proves a vital component for understanding the development of realisations, and thus permits an effective investigation of parameter

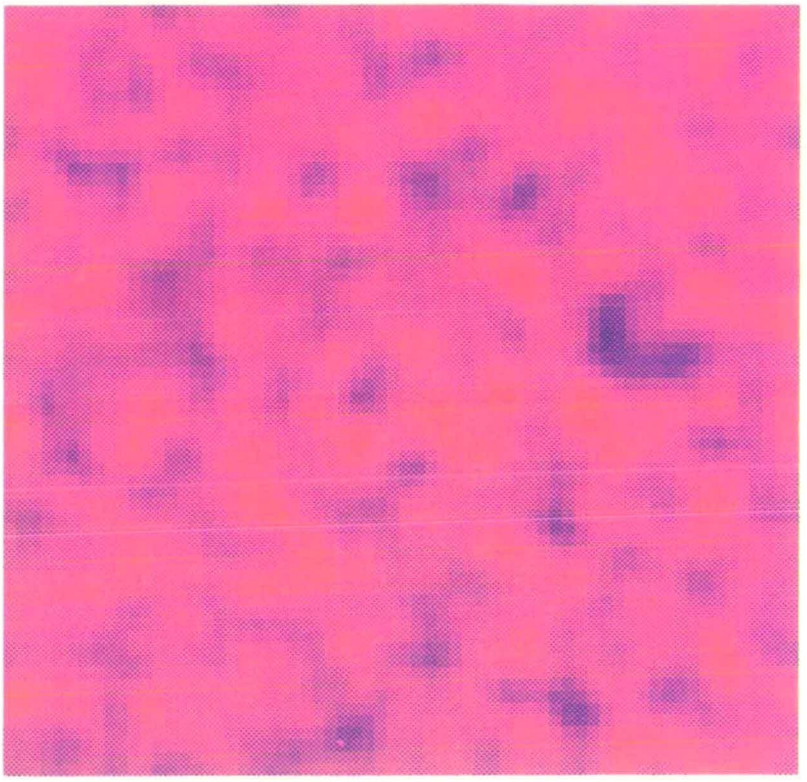

Figure 3.29: Extreme-long wavelength waves in two dimensions
A two-reactant spatial reaction system on a 64×64 location domain with periodic boundary conditions. System parameters are chosen to produce stationary waves with extreme-long wavelength — $\mu = \nu = 0.25, a = b = c = d = 1.0$. Initial perturbations consist of randomly distributed 1% changes to all reactant populations.

space. All of the graphical results presented in this section have been produced directly from the standard graphics interface provided on the parallel hardware, and have produced realisations that complement the fine graphical results detailed in an earlier work by Nagorcka & Mooney [1985], and the excellent review provided by Murray [1989].

3.6.1 Limiting Case Behaviour

In Chapter 2 we discussed the various limiting cases of the linearised two-reactant spatial reaction system in one dimension. We noted earlier Turing [1952] had predicted that the case of extreme-long wavelength waves could produce two-dimensional behaviour of more interest than the one-dimensional studies. However he was, of course, unable to study such systems in the manner we can today by using advanced computing technology.

If we recall the standard (linearised) parameter set for the extreme-long wavelength scenario (see Section 3.2), we used reactant migration rates of $\mu = \nu = 1/4$, cross-interaction coefficients $b = c = 1$, and self-interaction coefficients $a = d = 1$. In the one-dimensional system these parameters produce a stationary kilter scenario where all cell populations tend to move together away from equilibrium. In Figure 3.29 we see the graphical output from a large spatial system in two-dimensions (64×64 locations) using these same parameters but with migration to the four nearest neighbour locations. The system initial conditions are small levels (1% of equilibrium population) of random perturbation added to equilibrium populations. We observe that the whole system does not now act in unison, but that different areas cluster around different population levels. We can still observe "long-wavelength" behaviour as the areas of similar activity cover large numbers of cells, but the system as a whole takes on a quite un-deterministic aspect — similarities can be drawn to the dappled animal-coat patterns suggested by Turing [1952] and extensively investigated by Murray [1981].

3.6.2 Finite Wavelength Behaviour

The second stage of our study of the two-dimensional system involves a translation of our finite wavelength system. Using a non-linear parameter set identical to that used for the realisations detailed in Section 3.3, we can produce a two-reactant system in which both reactants interact exactly as in the one-dimensional scenario, but can now migrate in four directions to the four nearest neighbour locations. In the graphical output from such a realisation we observe the clear presence of spatial waves spreading from a single perturbation to dominate the steady-state solution (see Figure 3.30). Note that we observe approximately the same number of waves along each axis of the two-dimensional model, as we expect to find from the one-dimensional system, upon which the parameter set is based.

However, we feel that we obtain even more interesting results for this system if we consider starting from an equilibrium state, but with very many small perturbations to the individual cell populations (as in the limiting case behaviour above). We are thus, in effect, adding a low level ($\sim 1\%$) of random noise to all reactant populations. If we take such a system (with non-linear interactions, and super-critical parameters) and run

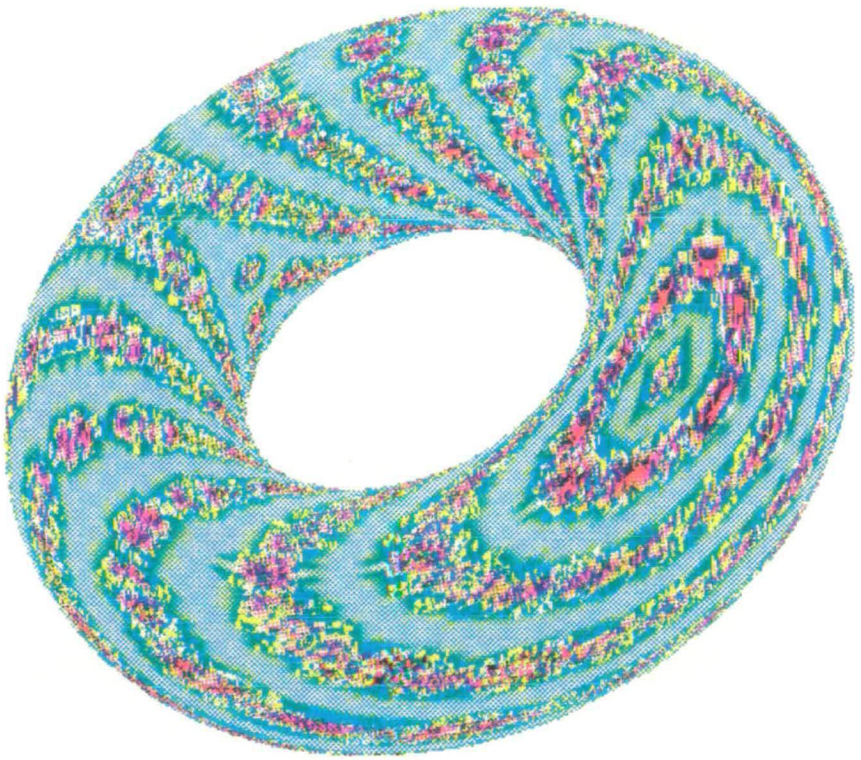

Figure 3.30: Spatial waves on the surface of a torus

This realisation output from a DAP-608 massively parallel supercomputer is manipulated by a Silicon Graphics workstation to produce the toroidal surface display corresponding to a two-dimensional two-reactant spatial reaction system with periodic boundary conditions. Using our standard non-linear interactions migration rates are chosen to give eight waves along each axis according to the linearised one-dimensional approximation. A populated equilibrium system is perturbed at a single point (visible to the centre-right of the torus) and we observe spatial waves radiating from this point. The colours in the display show the population of reactant X at each of 64×64 locations; dark colours show high population cells, lighter colours represent low populations.

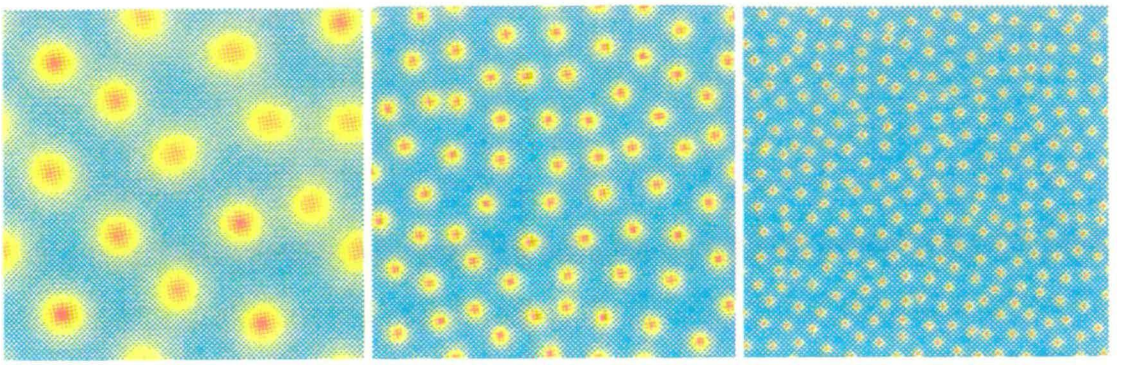

Figure 3.31: Deterministic hot-spots from random initial populations
The realisation output from a Connection Machine CM-200 massively parallel supercomputer is recorded directly from an X-windows workstation display. These images show a two-dimensional spatial reaction system using *noisy* initial equilibrium population. The migration rates were chosen to produce four, eight and sixteen waves in each dimension, respectively from left to right in the displays. The colours in the display show the population of reactant X at each of 64×64 locations; red colours show high population cells; light blue represents populations close to, or below, equilibrium levels; and yellow coloured locations are those with populations slightly above the equilibrium level.

a realisation forward in time, we observe the production of distinct *hot-spots* of reactant population (see Figure 3.31) similar to those identified over thirty years ago in the ground-breaking work of Maynard-Smith and Sondhi [1961] on embryo development. We find that the spatial distribution and abundance of these high-population points is controlled directly by the migration rate, set by the chosen value of system periodicity, $\lambda' = s_0/N$, which thus directly controls the desired number of waves (s_0) given the fixed size of the system, N.

The graphs in Figure 3.31 show the variation of hot-spot abundance for varying values of λ' and hence various numbers of intended waves. We observe that from each random starting situation all scenarios settle to a stable structure with approximately the expected number of wave-peaks (or hot-spots) along any axis taken through the spatial domain. Figure 3.31 clearly shows that strong local concentrations can be produced in two-dimensional systems with stable asymptotic states. From a random initial configuration, we cannot predict the exact location of hot-spots, however we can closely specify their distribution, size and abundance. We feel that such systems relate to early stages of development in embryos and genetic work. In particular, this is a much stronger result than that of Bard & Lauder [1974] and Bunow *et al* [1980] who had great doubts about the usefulness of such systems. We believe that we have shown a straightforward system

for the production of realistic natural effects, by simply considering a system with non-linear local interactions coupled with a variation in reactant migration (or diffusion) rate. We thus avoid the complications of potentially artificial *space dependent* diffusion terms (see Maini *et al* [1992]) and interaction terms (see Hunding [1990]).

We have obviously just touched on the potential scope of these two-dimensional studies. The initial conditions that we have used here have given the first indication of where we feel that further analytic and empirical study is required. The random nature of these starting perturbations is, we believe, linked inherently to the production of results that show *natural* effects. The next stage of our investigation therefore turns to the study of systems that operate in a stochastic mode throughout their development.

4

Mathematical Analysis of Stochastic Spatial Reaction Systems

At bottom, the theory of probability is only common sense reduced to calculation.

– Pierre Simon de Laplace

4.1 An Introduction to Stochastic Models

The next stage of our search for mathematical understanding of spatial reaction systems is to study the stochastic representation of the system equations. In particular we are interested in obtaining a measure of the spatial and temporal variation for each of our reactant populations, as this will lead us to some theoretical understanding of the behaviour of stochastic spatial reaction systems. A large number of examples can be drawn from nature to show the importance of random fluctuations in the environment in order to produce *realistic* behaviour. We meet the idea of *chance effects* being of importance to theoretical population biology in the work of Kendall [1949]. He references Feller [1939] as the first to define mathematically a stochastic birth-death population scenario.

The importance of considering complex and stochastic models has been well known for some time, and efforts have been made in many fields to formulate stochastic mathematical analyses. Initial work by Einstein into Brownian Motion [1905] introduced many of the principles of stochastic modelling of natural phenomena; a study of this contribution is given by Gardiner [1985]. Further efforts into stochastic modelling theory have been particularly strong in the disciplines of Ecology and Biology, where the interface to the real world is perhaps more immediate. In addition, interest has also grown in connection with the study of social processes, e.g. modelling the spread of news, rumours and ideas within human society (see Bartholomew [1982]). In an excellent position paper from over 35 years ago Bartlett [1957] expressed the need for stochastic models (of global populations) in order to produce mathematical results that can mimic experimental observations. As an example, Bartlett highlights the experimental results of Gause [1935] that require a such a model to exhibit sustained oscillations rather than the damped variety produced by standard deterministic techniques. In their fine recent review paper, Mollison, Isham and Grenfell [1994] leave no doubt as to the importance of stochastic representations of epidemic models. We feel that their considerations of discrete stochastic phenomena and population heterogeneity translate exactly to our general spatial reaction systems.

One of our ultimate aims in our work is to produce simulations of evolutionary processes. These necessarily include random mutation of some attributes of system reactants in order to observe the evolution of *fitter* solutions. However, even for the more straight-

forward stochastic simulations detailed in Chapter 5, the results we obtain may differ substantially from those predicted by a purely deterministic analysis of the governing system equations. We believe that it is also necessary to move away from non-spatial (global) populations as we search for accurate representations of real-world systems. Although the added complication of spatial dimensions makes solving stochastic system equations much more difficult, even greatly simplified equations can provide interesting results such as damped one- and two-dimensional population waves (see Bailey [1968]). Therefore the analysis we develop later in this chapter will tackle a general stochastic *spatial* reaction system.

4.1.1 Mathematical Formulation of Stochastic Systems

There are a number of methods available to construct mathematical descriptions of stochastic systems, and clear differentiation must be made between exact stochastic (or *probabilistic*) representations, and approximate techniques that typically treat the stochastic part of the system as an addition to a deterministic model. Probably the best known approach to generating exact stochastic equations is to consider such a system as a Markov process, and thereby generate the Kolmogorov differential equations from the probabilities of transfer between the available states. This approach is formulated as the *random variable* technique by Bailey [1964], and has been applied by the same author to the stochastic simple epidemic [1950, 1963]. The random variable technique has also been applied to a subject closer to our own interest by Renshaw [1973] which deals with stochastic versions of spatial population processes for a single reactant. It is generally accepted (see Chapter 1 of Gardiner [1985]) that equations resulting from this type of derivation prove very difficult to solve, even for simple single reactant systems. We must typically resort to approximations in terms of linearisation of interaction functions, or the use of specific initial conditions. This difficulty is greatly increased as we consider more complex systems such as our coupled two-reactant model. Thus full probabilistic solution of a spatial system with non-linear stochastic interactions, has to be regarded as intractable with current mathematical methods.

However, there is now an established model for approximating stochastic systems based on considering the system stochasticity to be external to a (well-understood) determ-

inistic model. We can think of this as simply the addition of *Gaussian noise* to the right hand side of a deterministic differential equation. Here "noise" describes an idealised, completely random process, with individual random variables being distributed Normally and independently. The first formulation of such a stochastic system was by Langevin in 1908 (see Gard [1988] as a reference to the original) to describe the motion of a particle in a random force field. This work was a continuation of the theoretical Physics drive towards understanding the Brownian motion problem that was initiated by Einstein. The Gaussian noise technique (often termed *stochastic linearisation*) has been popularised in the field of Ecology and Epidemiology by Bartlett in his comprehensive text books [1955, 1960]. In those fields it has been used to determine the expected values of large populations that remain close to equilibrium values, thus allowing the possibility of reactant extinction to be ignored.

The stochastic linearisation approximation provides us with a technique for producing solutions to the stochastic version of many deterministic models, although we must remember that the full behaviour of a stochastic model is not determined entirely by the expectation values of the random variables. Strictly we should consider the whole probability distribution of all variables at each time interval to achieve such a result (see Smith & Mead [1980]). However the Gaussian noise technique has proved to be an excellent (and tractable) approximation for much biological and ecological work with stochastic difference equations, and we shall generally restrict ourselves to this approach. It is possible, however, to produce stochastic differential equations for continuous time solutions, and the stochastic linearisation technique was formalised into a theory of stochastic differential equations by Ito [1951], with an alternative approach proposed later by Stratonovich [1966]. Such formulations are now well established in the engineering field, and recent texts (see Gard [1988]) also acknowledge their increasing application in Biology and Ecology, with the work of Turelli [1977] being a fine example of the union of the two disciplines. This same author has performed further work [1978] that shows that the stability criterion for exact stochastic models using stochastic differential equation techniques is inherently different from that for deterministic scenarios.

It is important to remember that there can be a substantial difference between the results we obtain from stochastic systems developed using the different techniques described in this section. If we regard pure deterministic modelling, and full probabilistic modelling,

as lying at the two opposite extremes of potential modelling techniques, then the Bartlett-type Gaussian noise methodology can be placed somewhere between the two. Although we will choose to concentrate on this technique for our work (for reasons of tractability of solution — Renshaw [1972] states that the full probabilistic equations for even a single species spatial model are immune to direct solution), it cannot be regarded as an exact representation of stochastic systems but rather as an approximation, and we must be ready to qualify our stochastic linearisation results accordingly. The best method for this qualification is to compare our results with the full stochastic simulations detailed later in Chapter 5.

4.2 Formulation of the One-Dimensional System

The stochastic equations for a general spatial reaction system in one dimension with periodic boundary conditions can be formulated by adding Bartlett-type Gaussian noise to the deterministic equations (2.1) with non-linear interaction functions (2.19) studied in Chapter 2. This allows us to develop expressions for the auto- and cross-covariance coefficients for two reactants on a ring of cells via a Fourier transform approach. We perform this analysis for both spatial and temporal correlations; the first provides us with information on the morphological structure of our stochastic reactant populations, whereas the second allows us to understand the stochastic development of reactant populations in time. The results allow us to find the typical *half-life* of locations with high concentration populations. This information is used to implement stochastic simulations efficiently on parallel computers, as detailed later in Chapters 6 and 7.

Recalling our general deterministic equations for a spatial reaction system with non-linear interactions on a ring of N cells containing populations of species X_r and Y_r where $r = 1$ to N, (i.e. (2.1) and (2.19)), we have

$$dX_r(t)/dt = X_r(t)[r_1 + a_1X_r(t) + b_1Y_r(t)] + \mu[X_{r+1}(t) - 2X_r(t) + X_{r-1}(t)] \quad (4.1)$$

$$dY_r(t)/dt = Y_r(t)[r_2 + a_2X_r(t) + b_2Y_r(t)] + \nu[Y_{r+1}(t) - 2Y_r(t) + Y_{r-1}(t)]. \quad (4.2)$$

To convert (4.1) and (4.2) into stochastic equations by allowing for random fluctuations in the values of X_r and Y_r, we add to each equation a site-dependent noise component

$\delta Z_r^X(t)$ and $\delta Z_r^Y(t)$. This is a spatial and multi-reactant extension to the technique popularised by Bartlett [1957] as discussed in Section 4.1.1. These random variables are independent with respect to time, site and to each other, and have mean values of zero, but non-zero variances as detailed below in Section 4.2.2. By introducing a small time increment δt we obtain

$$\begin{aligned} X_r(t+\delta t) &= X_r(t) + X_r(t)\delta t[r_1 + a_1 X_r(t) + b_1 Y_r(t)] \\ &+ \mu\delta t[X_{r+1}(t) - 2X_r(t) + X_{r-1}(t)] + \delta Z_r^X(t) \qquad (4.3) \\ Y_r(t+\delta t) &= Y_r(t) + Y_r(t)\delta t[r_2 + a_2 X_r(t) + b_2 Y_r(t)] \\ &+ \nu\delta t[Y_{r+1}(t) - 2Y_r(t) + Y_{r-1}(t)] + \delta Z_r^Y(t). \qquad (4.4) \end{aligned}$$

These discrete time equations can be compared to the stochastic differential equations produced by letting $\delta t \to 0$ (see Gard [1988], page 182) to give

$$dX_r = X_r(r_1 + a_1 X_r + b_1 Y_r)dt + \mu dt(X_{r+1} - 2X_r + X_{r-1}) + g_1 X_r dW_1(t) \quad \text{and}$$

$$dY_r = Y_r(r_2 + a_2 X_r + b_2 Y_r)dt + \nu dt(Y_{r+1} - 2Y_r + Y_{r-1}) + g_2 Y_r dW_2(t)\ ,$$

where the $W_j(t)$ are independent Weiner processes, and the g_j are constant coefficients.

4.2.1 Linearisation

The direct solution of the full non-linear equations (4.1) and (4.2) has proved intractable in the deterministic case, and it is therefore highly unlikely that we could produce a complete solution of the more complex non-linear stochastic form shown in (4.3) and (4.4). We therefore look towards some simplification of the system. If we can assume (as in Chapter 2) that the populations X_r and Y_r lie close to some equilibrium state (X^*, Y^*), and that the system consists of small perturbations $x_r(t)$ and $y_r(t)$ around this value, we may introduce the linearisations

$$X_r(t) = X^*[1 + x_r(t)] \quad \text{and} \quad Y_r(t) = Y^*[1 + y_r(t)]\ . \qquad (4.5)$$

These are similar to the approximations used for the deterministic case (2.2), and give identical results to first-order calculations, although this equality does not hold to higher

orders. This particular representation is preferred here as it introduces perturbations as a proportion of the equilibrium population, rather than in absolute terms, thereby reducing the complexity of results for the general case when $X^* \neq Y^*$. By substituting (4.5) into (4.3) and (4.4) we obtain

$$\begin{aligned}
X^*[1+x_r(t+\delta t)] &= X^*[1+x_r(t)] + \delta Z_r^X(t) \\
&+ X^*[1+x_r(t)]\delta t(r_1 + a_1X^*[1+x_r(t)] + b_1Y^*[1+y_r(t)]) \\
&+ \mu\delta t(X^*[1+x_{r+1}(t)] - 2X^*[1+x_r(t)] + X^*[1+x_{r-1}(t)]) \\
Y^*[1+y_r(t+\delta t)] &= Y^*[1+y_r(t)] + \delta Z_r^Y(t) \\
&+ Y^*[1+y_r(t)]\delta t(r_2 + a_2X^*[1+x_r(t)] + b_2Y^*[1+y_r(t)]) \\
&+ \nu\delta t(Y^*[1+y_{r+1}(t)] - 2Y^*[1+y_r(t)] + Y^*[1+y_{r-1}(t)]) \ .
\end{aligned}$$

We have seen previously that at equilibrium (X^*, Y^*) our general interactions can be simplified using the relations (2.21), and this allows us to reduce the equations above to linearised stochastic equations in the variables x_r and y_r, i.e. the site-dependent perturbations expressed as a ratio of the equilibrium populations. Dividing the above equations by X^* and Y^* respectively, regarding all terms of higher than single order in x_r and y_r as negligible (since we assume departures from equilibrium to be *small*), then yields

$$\begin{aligned}
x_r(t+\delta t) - x_r(t) &= [a_1X^*x_r(t) + b_1Y^*y_r(t)]\delta t \\
&+ \mu\delta t[x_{r+1}(t) - 2x_r(t) + x_{r-1}(t)] + \delta Z_r^X(t)/X^* \qquad (4.6) \\
y_r(t+\delta t) - y_r(t) &= [a_2X^*x_r(t) + b_2Y^*y_r(t)]\delta t \\
&+ \nu\delta t[y_{r+1}(t) - 2y_r(t) + y_{r-1}(t)] + \delta Z_r^Y(t)/Y^* \ . \qquad (4.7)
\end{aligned}$$

The assumption of small departures from equilibrium is known to be inaccurate as we have many simulated results of populations moving away from equilibrium by amounts far greater than the equilibrium population itself, and at exponential rates. However, since it is generally the initial movements away from the equilibrium state that determine any final system morphology, we can use the above simplifications to find these important initial small perturbations. In addition, the recent results of Renshaw [1991] suggest that this linearisation technique is surprisingly accurate, even when populations have moved substantially away from equilibrium.

4.2.2 Probability Analysis of the General System

Let us now look at the probabilities of various integer changes in the reactant populations X_r and Y_r, during a time interval δt. This analysis will lead us to a measure of the expectation and variance/covariance of the random variables $\delta Z_r^X(t)$ and $\delta Z_r^Y(t)$. If we assume that all populations are near the equilibrium values (X^*, Y^*) then we can write down the probabilities for the evolution of the system within each cell. It can be seen that this leads to an overall migration effect of zero, since we are in a global equilibrium state. For a system with positive instability we know that the final state is often not in global equilibrium, but in some stationary *phase-locked* state with local equilibria in each cell. However, in order to produce values for the variances of our random variables $\delta Z_r^X(t)$ and $\delta Z_r^Y(t)$ we must accept that for early movements away from equilibrium the global conditions hold true.

Since we are dealing with our general non-linear interaction equations, with three interaction coefficients for each species, we must develop some general system to express simply the probability of positive and negative integer changes in the reactant population. Let us therefore borrow some terminology from ecological studies, and consider probabilities for the *birth* and *death* of a reactant. This introduces a conceptual problem when one considers the scenario of, for example, interacting chemicals. However, this problem is only one of terminology, and a straightforward choice of certain coefficients to represent death (i.e. a negative integer change), and others to represent birth (i.e. a positive integer change) will suffice for our purposes, since the mathematical effects will be controlled by the sign of the individual coefficients. Such a pragmatic choice can therefore be made based upon the sign of the coefficients as used in our generic examples in Chapter 3 — see Table 3.1. We can thus choose the b_j coefficients to represent birth, and the a_j and r_j to govern death within each reactant type j. Again, it must be stressed that we are not actually dealing with a birth-death scenario, but rather with our general spatial reaction system. This arbitrary choice forces us to negate the probability value for deaths in order to retain full generality in our system, both mathematically and conceptually. Thus the probability values detailed below apply to the specific scenario of positive b_j values and negative a_j and r_j values denoting *birth* and *death* respectively, hence

$$
\begin{aligned}
\Pr[X_r \text{ birth}] &= \Pr[\delta Z_r^X(t) = +1] &&= b_1 X^* Y^* \delta t \ \ (b_1 \geq 0) \\
\Pr[Y_r \text{ birth}] &= \Pr[\delta Z_r^Y(t) = +1] &&= b_2 (Y^*)^2 \delta t \ \ (b_2 \geq 0) \\
\Pr[X_r \text{ death}] &= \Pr[\delta Z_r^X(t) = -1] &&= -(r_1 + a_1 X^*) X^* \delta t \ \ (r_1, a_1 \leq 0) \\
\Pr[Y_r \text{ death}] &= \Pr[\delta Z_r^Y(t) = -1] &&= -(r_2 + a_2 X^*) Y^* \delta t \ \ (r_2, a_2 \leq 0) \\
\Pr[X_r \text{ migrate}] &= \Pr[\delta Z_r^X(t) = +1] &&= \mu \delta t (X^* - 2X^* + X^*) = 0 \\
\Pr[Y_r \text{ migrate}] &= \Pr[\delta Z_r^Y(t) = +1] &&= \nu \delta t (Y^* - 2Y^* + Y^*) = 0.
\end{aligned}
$$

If we now calculate the expectation value of the two random variables we can confirm that they equal zero, as follows, using the relations (2.21) to simplify where necessary

$$
\begin{aligned}
\mathrm{E}\{\delta Z_r^X(t)\} &= (+1)\Pr[\delta Z_r^X(t) = +1] + (-1)\Pr[\delta Z_r^X(t) = -1] \\
&= (+1)(b_1 X^* Y^* \delta t) + (-1)(-X^* \delta t (r_1 + a_1 X^*)) \\
&= X^* \delta t (r_1 + a_1 X^* + b_1 Y^*) = 0 \\
\mathrm{E}\{\delta Z_r^Y(t)\} &= (+1)\Pr[\delta Z_r^Y(t) = +1] + (-1)\Pr[\delta Z_r^Y(t) = -1] \\
&= (+1)(b_2 (Y^*)^2 \delta t) + (-1)(-Y^* \delta t (r_2 + a_2 X^*)) \\
&= Y^* \delta t (r_2 + a_2 X^* + b_2 Y^*) = 0 \ .
\end{aligned}
$$

We can also calculate the variance of each of the random variables as follows, again using the relations (2.21) to simplify where necessary.

$$
\begin{aligned}
\mathrm{Var}\{\delta Z_r^X(t)\} &= \mathrm{E}\{\delta Z_r^X(t)^2\} - \mathrm{E}\{\delta Z_r^X(t)\}\mathrm{E}\{\delta Z_r^X(t)\} \\
&= (+1)^2 \Pr[\delta Z_r^X(t) = +1] + (-1)^2 \Pr[\delta Z_r^X(t) = -1] \\
&= b_1 X^* Y^* \delta t - r_1 X^* \delta t - a_1 (X^*)^2 \delta t \\
&= 2 b_1 X^* Y^* \delta t \qquad (4.8) \\
\mathrm{Var}\{\delta Z_r^Y(t)\} &= \mathrm{E}\{\delta Z_r^Y(t)^2\} - \mathrm{E}\{\delta Z_r^Y(t)\}\mathrm{E}\{\delta Z_r^Y(t)\} \\
&= (+1)^2 \Pr[\delta Z_r^Y(t) = +1] + (-1)^2 \Pr[\delta Z_r^Y(t) = -1] \\
&= b_2 (Y^*)^2 \delta t - r_2 Y^* \delta t - a_2 X^* Y^* \delta t \\
&= 2 b_2 (Y^*)^2 \delta t. \qquad (4.9)
\end{aligned}
$$

It can also be shown that since the random variables are independent, their covariance at a particular site r and time t is of order δt^2, and so can be ignored under our current

approximations. That is

$$\begin{aligned}
\mathrm{Cov}\{\delta Z_r^X(t), \delta Z_r^Y(t)\} &= \mathrm{E}\{\delta Z_r^X(t)\delta Z_r^Y(t)\} - \mathrm{E}\{\delta Z_r^X(t)\}\mathrm{E}\{\delta Z_r^Y(t)\} \\
&= (+1)^2(\Pr[\delta Z_r^X(t) = +1] \times \Pr[\delta Z_r^Y(t) = +1]) + \\
&\quad (-1)^2(\Pr[\delta Z_r^X(t) = -1] \times \Pr[\delta Z_r^Y(t) = -1]) + \\
&\quad (+1)(-1)(\Pr[\delta Z_r^X(t) = +1] \times \Pr[\delta Z_r^Y(t) = -1]) + \\
&\quad (+1)(-1)(\Pr[\delta Z_r^X(t) = -1] \times \Pr[\delta Z_r^Y(t) = +1]) \\
&= X^*Y^*\delta t^2[b_1b_2(Y^*)^2 + (r_1 + a_1X^*)(r_2 + a_2X^*) \\
&\quad + b_1Y^*(r_2 + a_2X^*) + b_2Y^*(r_1 + a_1X^*)] \\
&= O(\delta t)^2 \ .
\end{aligned}$$

In a similar way we can show that the other spatial and temporal covariance values for our independent random variables (e.g. the covariance for site r at times t and s say, or that for sites r and q at time t) can be taken as zero, since they too are of order δt^2.

4.2.3 Expectation Values of the Linear System

Since the expectation values for our random variables can be taken as zero, we can produce a tractable linearised solution for the stochastic mean perturbation values. Taking expectation values of the linear stochastic equations (4.6) and (4.7) gives

$$dm_r(t)/dt = a_1X^*m_r(t) + b_1Y^*n_r(t) + \mu(m_{r+1}(t) - 2m_r(t) + m_{r-1}(t)) \qquad (4.10)$$

$$dn_r(t)/dt = a_2X^*m_r(t) + b_2Y^*n_r(t) + \nu(n_{r+1}(t) - 2n_r(t) + n_{r-1}(t)), \qquad (4.11)$$

where $m_r(t) = \mathrm{E}\{x_r(t)\}$ and $n_r(t) = \mathrm{E}\{y_r(t)\}$. Thus the equations for the linear stochastic mean perturbations reduce exactly to the linearised deterministic system detailed in Equations (2.3) and (2.4). We can therefore use the results of Turing's initial analysis to detail the time evolution of the expected values of the stochastic population perturbations.

4.3 Covariance Analysis Using Fourier Transforms

Having produced linearised stochastic equations (4.6) and (4.7) that describe the behaviour of site-dependent perturbations away from a global equilibrium state, we now need to find a mechanism for determining the spatial and temporal correlations that exist between these perturbations. In order to achieve this we can build upon a technique first introduced by Renshaw [1984] for the analysis of competition spectra in plant-growth, and used more recently by the same author [1994] for a spatial system with a single species undergoing linear self-interaction. We will now extend and adapt this approach to our general two-species scenario, where we must consider two auto-covariance functions (one for each species), as well as a measure of cross-covariance. Therefore, following Renshaw [1994], let us introduce a new variable, u, to represents the cell *lag* between any two instances of the stochastic system. We can thus write two lag-adapted versions of equations (4.6) and (4.7) by replacing r with $r+u$, namely

$$\begin{aligned} x_{r+u}(t+\delta t) - x_{r+u}(t) &= [a_1 X^* x_{r+u}(t) + b_1 Y^* y_{r+u}(t)]\delta t \qquad (4.12) \\ &+ \mu\delta t[x_{r+u+1}(t) - 2x_{r+u}(t) + x_{r+u-1}(t)] + \delta Z^X_{r+u}(t)/X^* \\ y_{r+u}(t+\delta t) - y_{r+u}(t) &= [a_2 X^* x_{r+u}(t) + b_2 Y^* y_{r+u}(t)]\delta t \qquad (4.13) \\ &+ \nu\delta t[y_{r+u+1}(t) - 2y_{r+u}(t) + y_{r+u-1}(t)] + \delta Z^Y_{r+u}(t)/Y^* \ . \end{aligned}$$

4.3.1 Introduction of Second-Order Moments

Let us introduce the following definitions for three second-order moments:

$$\begin{aligned} \sigma^X_u(t) &= \mathrm{E}\{[x_r(t) - m_r(t)][x_{r+u}(t) - m_{r+u}(t)]\} \\ &= \mathrm{E}\{x_r(t)x_{r+u}(t)\} - m_r(t)m_{r+u}(t) \\ \sigma^Y_u(t) &= \mathrm{E}\{[y_r(t) - n_r(t)][y_{r+u}(t) - n_{r+u}(t)]\} \\ &= \mathrm{E}\{y_r(t)y_{r+u}(t)\} - n_r(t)n_{r+u}(t) \\ \sigma^{XY}_u(t) &= \mathrm{E}\{[x_r(t) - m_r(t)][y_{r+u}(t) - n_{r+u}(t)]\} \\ &= \mathrm{E}\{x_r(t)y_{r+u}(t)\} - m_r(t)n_{r+u}(t) \ . \end{aligned}$$

We can simplify these expressions if we consider our system to have started from a uniform equilibrium state, i.e. $X_r(0) \equiv X^*$ and $Y_r(0) \equiv Y^*$. Thus the initial expected

perturbation values $m_r(0) = n_r(0) = 0$ for all r, and hence, provided our linearisation approximation remains valid, equations (4.10) and (4.11) show that all future expected values for perturbations $m_r(t) = n_r(t) = 0$ for all r and all t. We can therefore write

$$\begin{aligned} \mathrm{E}\{x_r(t)x_{r+u}(t)\} &= \sigma_u^X(t), \\ \mathrm{E}\{y_r(t)y_{r+u}(t)\} &= \sigma_u^Y(t) \quad \text{and} \\ \mathrm{E}\{x_r(t)y_{r+u}(t)\} &= \sigma_u^{XY}(t) \ . \end{aligned} \tag{4.14}$$

We must now look for mechanisms to produce expressions for the expectation values on the left-hand side of the above equations. Taking the X reactant first, multiplying (4.6) and (4.12) gives

$$\begin{aligned} x_r(t+\delta t)x_{r+u}(t+\delta t) &= x_r(t)x_{r+u}(t) + \delta Z_r^X(t)\delta Z_{r+u}^X(t)/(X^*)^2 \\ &+\delta t[2a_1X^*x_r(t)x_{r+u}(t) + b_1Y^*(x_r(t)y_{r+u}(t) + x_{r+u}(t)y_r(t)) + \mu(x_r(t)x_{r+u+1}(t) \\ &+x_r(t)x_{r+u-1}(t) + x_{r+u}(t)x_{r+1}(t) + x_{r+u}(t)x_{r-1}(t) - 4x_r(t)x_{r+u}(t))] \end{aligned}$$

where we have already ignored all terms in δt^2 (as negligible) and all terms that are first-order in $\delta Z_r^X(t)$, as we know that these will equate to zero when we perform the next step of taking expected values. On taking expectations and using the relations (4.14) we can write

$$\begin{aligned} \sigma_u^X(t+\delta t) &= \sigma_u^X(t) + \delta t[2a_1X^*\sigma_u^X(t) + b_1Y^*(\sigma_u^{XY}(t) + \sigma_{-u}^{XY}(t))] \\ &+ \mu\delta t[2\sigma_{u+1}^X(t) + 2\sigma_{u-1}^X(t) - 4\sigma_u^X(t)] + 2b_1Y^*\delta t\delta(u;0)/X^*, \end{aligned}$$

where $\delta(u;0)$ is the Kronecker delta function which equals unity when $u = 0$ and is zero otherwise. This allows the incorporation of the variance of the random variable ($\delta Z_r^X(t) = 2b_1X^*Y^*\delta t$), which is only non-zero (or of order δt) when $u = 0$, since multiplication by the Kronecker delta function, $\delta(u;0)$, produces a term that is always zero unless $u = 0$.

After division by δt and letting $\delta t \to 0$ we obtain the following differential equation for the second-order moment $\sigma_u^X(t)$, i.e. the auto-covariance of perturbations in population X at cell lag u:

$$d\sigma_u^X(t)/dt = (2a_1X^* - 4\mu)\sigma_u^X(t) + b_1Y^*(\sigma_u^{XY}(t) + \sigma_{-u}^{XY}(t))$$

$$+ \quad 2\mu(\sigma^X_{u+1}(t) + \sigma^X_{u-1}(t)) + 2b_1 Y^* \delta(u;0)/X^* \quad . \tag{4.15}$$

We can now perform a similar analysis for the Y reactant by multiplying (4.7) and (4.13) to obtain

$$\begin{aligned} y_r(t+\delta t)y_{r+u}(t+\delta t) &= y_r(t)y_{r+u}(t) + \delta Z^Y_r(t)\delta Z^Y_{r+u}(t)/(Y^*)^2 \\ &+\delta t[2b_2 Y^* y_r(t)y_{r+u}(t) + a_2 X^*(y_r(t)x_{r+u}(t) + y_{r+u}(t)x_r(t)) + \nu(y_r(t)y_{r+u+1}(t) \\ &+y_r(t)y_{r+u-1}(t) + y_{r+u}(t)y_{r+1}(t) + y_{r+u}(t)y_{r-1}(t) - 4y_r(t)y_{r+u}(t))] \ , \end{aligned}$$

where we have again ignored all terms in δt^2 and all terms that are first-order in $\delta Z^Y_r(t)$. By considering the expected value of all the terms in this equation, and using (4.14) and the variance values from (4.9), we can write

$$\begin{aligned} \sigma^Y_u(t+\delta t) &= \sigma^Y_u(t) + \delta t[2b_2 Y^* \sigma^Y_u(t) + a_2 X^*(\sigma^{XY}_u(t) + \sigma^{XY}_{-u}(t))] \\ &+ \nu\delta t[2\sigma^Y_{u+1}(t) + 2\sigma^Y_{u-1}(t) - 4\sigma^Y_u(t)] + 2b_2\delta t\delta(u;0) \ . \end{aligned}$$

After division by δt and letting $\delta t \to 0$ we obtain a second differential equation, on this occasion for the second-order moment $\sigma^Y_u(t)$, i.e. the auto-covariance of perturbations in population Y at cell lag u,

$$\begin{aligned} d\sigma^Y_u(t)/dt &= (2b_2 Y^* - 4\nu)\sigma^Y_u(t) + a_2 X^*(\sigma^{XY}_u(t) + \sigma^{XY}_{-u}(t)) \\ &+ 2\nu(\sigma^Y_{u+1}(t) + \sigma^Y_{u-1}(t)) + 2b_2\delta(u;0) \ . \end{aligned} \tag{4.16}$$

Finally, we can produce a differential equation for the cross-covariance of perturbations in the populations X and Y at cell lag u by considering the multiplication of equations (4.6) and (4.13). By ignoring all terms in δt^2, first-order in $\delta Z^X_r(t)$ and $\delta Z^Y_r(t)$, and $\delta Z^X_r(t)\delta Z^Y_r(t)$ cross-terms, we obtain

$$\begin{aligned} x_r(t+\delta t)y_{r+u}(t+\delta t) &= x_r(t)y_{r+u}(t) + \delta t[a_2 X^* x_r(t)x_{r+u}(t) \\ &+b_2 Y^* x_r(t)y_{r+u}(t) + a_1 X^* y_{r+u}(t)x_r(t) + b_1 Y^* y_r(t)y_{r+u}(t)] \\ &+\nu\delta t[x_r(t)y_{r+u+1}(t) + x_r(t)y_{r+u-1}(t) - 2x_r(t)y_{r+u}(t)] \\ &+\mu\delta t[x_{r+1}(t)y_{r+u}(t) + x_{r-1}(t)y_{r+u}(t) - 2x_r(t)y_{r+u}(t)] \ , \end{aligned}$$

which yields

$$\begin{aligned} d\sigma_u^{XY}(t)/dt &= a_2X^*\sigma_u^X(t) + b_1Y^*\sigma_u^Y(t) + [a_1X^* + b_2Y^* - 2(\mu+\nu)]\sigma_u^{XY}(t) \\ &+ (\mu+\nu)[\sigma_{u+1}^{XY}(t) + \sigma_{u-1}^{XY}(t)] \ . \end{aligned} \tag{4.17}$$

4.3.2 Fourier Transform Representation

In order to solve this system of three coupled differential equations for the system covariances — (4.15), (4.16) and (4.17), let us follow the lead provided by Renshaw [1994] where he considers the single reactant system with periodic boundary conditions, i.e. on a ring of cells. This approach is also inherently tied to the initial studies of Turing [1952] covered earlier in Chapter 2, who suggested the use of Fourier transforms to solve this type of periodic system.

Let us define a Fourier transform of the X-perturbation covariance as

$$\Phi_\theta^X(t) = \frac{1}{N}\sum_{u=0}^{N-1}\sigma_u^X(t)e^{2\pi iu\theta/N}, \tag{4.18}$$

which has the standard inverse transform

$$\sigma_u^X(t) = \sum_{\theta=0}^{N-1}\Phi_\theta^X(t)e^{-2\pi iu\theta/N} \ . \tag{4.19}$$

Similarly we can define transforms for the Y- and XY-covariances as

$$\Phi_\theta^Y(t) = \frac{1}{N}\sum_{u=0}^{N-1}\sigma_u^Y(t)e^{2\pi iu\theta/N} \quad \text{and} \quad \Phi_\theta^{XY}(t) = \frac{1}{N}\sum_{u=0}^{N-1}\sigma_u^{XY}(t)e^{2\pi iu\theta/N} \tag{4.20}$$

respectively, with inverse transforms

$$\sigma_u^Y(t) = \sum_{\theta=0}^{N-1}\Phi_\theta^Y(t)e^{-2\pi iu\theta/N} \quad \text{and} \quad \sigma_u^{XY}(t) = \sum_{\theta=0}^{N-1}\Phi_\theta^{XY}(t)e^{-2\pi iu\theta/N} \ . \tag{4.21}$$

If we now multiply (4.15) by $e^{2\pi iu\theta/N}$ and sum all terms from $u = 0$ to $N-1$, then

division by N gives

$$\frac{1}{N}\sum_{u=0}^{N-1}\frac{d\sigma_u^X(t)}{dt}e^{2\pi iu\theta/N} = \frac{2a_1X^* - 4\mu}{N}\sum_{u=0}^{N-1}\sigma_u^X(t)e^{2\pi iu\theta/N} \tag{4.22}$$
$$+\frac{b_1Y^*}{N}\sum_{u=0}^{N-1}\sigma_u^{XY}(t)e^{2\pi iu\theta/N} + \frac{b_1Y^*}{N}\sum_{u=0}^{N-1}\sigma_{-u}^{XY}e^{2\pi iu\theta/N} + \frac{2\mu}{N}\sum_{u=0}^{N-1}\sigma_{u+1}^{X}e^{2\pi iu\theta/N}$$
$$+\frac{2\mu}{N}\sum_{u=0}^{N-1}\sigma_{u-1}^{X}e^{2\pi iu\theta/N} + \frac{2b_1Y^*}{NX^*}\sum_{u=0}^{N-1}\delta(u;0)e^{2\pi iu\theta/N} \quad .$$

Using the Fourier transforms defined above we can simplify this expression substantially to give the following differential equation for the transform of the X-perturbation auto-covariance,

$$\frac{d\Phi_\theta^X(t)}{dt} = 2\left(a_1X^* - 4\mu\sin^2\frac{\pi\theta}{N}\right)\Phi_\theta^X(t) + 2b_1Y^*\Phi_\theta^{XY}(t) + \frac{2b_1Y^*}{NX^*} \quad . \tag{4.23}$$

To obtain (4.23) it must be noted that three features of the system have been exploited.

1. Since we are dealing with a system with periodic boundary conditions, we can define a *circularity condition* for the system as

$$\sigma_{u+pN} = \sigma_u \text{ for all } u = 0,\ldots,N-1 \text{ and } \infty < p < \infty \ ,$$

for any integer p. This condition allows us to equate $\sum_{u=0}^{N-1}\sigma_u$ with $\sum_{u=0}^{N-1}\sigma_{-u}$, which removes the complexity of any $-u$ covariance terms. In addition, it also allows us to equate arithmetic progressions such as $\sum_{u=0}^{N-1}\sigma_u$ and $\sum_{u=0}^{N-1}\sigma_{u\pm1}$, although in this case we must take care to make the necessary changes to the exponential term within the summation, e.g.

$$\sum_{u=0}^{N-1}\sigma_{u\pm1}^{X}e^{2\pi iu\theta/N} = \Phi_\theta^X e^{\mp 2\pi i\theta/N} \quad .$$

2. We can simplify the final term in Equation (4.22) since

$$\frac{1}{N}\sum_{u=0}^{N-1}\frac{2b_1Y^*}{X^*}\delta(u;0)e^{2\pi iu\theta/N} = \frac{2b_1Y^*}{NX^*} \quad .$$

3. The final simplification is a straightforward use of the trigonometric relation

$$e^{2i\zeta} + e^{-2i\zeta} = 2\cos 2\zeta = 2(1 - 2\sin^2 \zeta),$$

which yields

$$2\mu\Phi_\theta^X(e^{2\pi i\theta/N} + e^{-2\pi i\theta/N} - 2) = -8\mu \sin^2 \frac{\pi\theta}{N} \ .$$

Using the above procedures on Equations (4.16) and (4.17) we obtain two further differential equations for the Fourier transforms of the Y- and XY-perturbation covariances. Multiplication of (4.16) and (4.17) by $e^{2\pi iu\theta/N}$, summation of all terms from $u = 0$ to $N - 1$, and then division by N gives

$$\frac{d\Phi_\theta^Y(t)}{dt} = 2\left(b_2Y^* - 4\nu \sin^2 \frac{\pi\theta}{N}\right)\Phi_\theta^Y(t) + 2a_2X^*\Phi_\theta^{XY}(t) + 2b_2/N \quad \text{and} \tag{4.24}$$

$$\frac{d\Phi_\theta^{XY}(t)}{dt} = a_2X^*\Phi_\theta^X(t) + b_1Y^*\Phi_\theta^Y(t) + (a_1X^* + b_2Y^* - 4(\mu + \nu)\sin^2 \frac{\pi\theta}{N})\Phi_\theta^{XY}(t). \tag{4.25}$$

4.3.3 Summary of System Approximations

In the previous section we have successfully worked through an analysis of the auto- and cross-covariance values (or second-order moments) for perturbations to the populations within a stochastic version of our generalised linear spatial reaction system in one (periodic) dimension. In Section 4.3.4 below we detail the resulting coupled differential equations. However, let us now review the approximations that have been made thus far, so as not to lose sight of the range of applicability of our results.

The first, and most important, approximation is that we can represent our stochastic system as of Bartlett type, with the stochasticity specified by an external random noise process added to the standard deterministic model. We know that this type of approach generally produces stochastic means in agreement with both the deterministic and full probabilistic representations, as we have shown for our system, but can produce variance values quite different from the full probabilistic scenario.

The second major approximation in use is the linearisation of the system interactions. Without this it would have been difficult to move our analysis past the initial non-linear stochastic equations (4.3) and (4.4), so we are forced to choose the linearisation approximation in order to solve the system analytically.

We also make a number of approximations and assumptions during the body of the analysis work in Sections 4.2 and 4.3. We assume that the initial state of our system is free of any perturbations, and our results therefore represent the population covariances that are produced purely by the fact that we have a stochastic system — i.e. the nature of the perturbations away from equilibrium are due solely to the noise variation in the system. In addition, we ignore all terms that are second-order in δt, an approximation that ties in with a study of the linearised system.

All of these simplifications to our system point us towards a study of small movements away from equilibrium, and thus typically to short times after $t = 0$. As stated earlier, we find that the typical morphology of spatial reaction systems is defined by these initial departures from equilibrium, and thus we feel that, despite this set of approximations and assumption, we have an analysis that provides important insight into the workings of stochastic spatial reaction systems. We detail later (in Chapter 5) how accurate these approximation prove to be.

4.3.4 Resulting Coupled Equations

The previous analysis has produced three coupled differential equations for the Fourier transform of the second-order moments of the perturbations away from an equilibrium state in our stochastic spatial reaction system — (4.23), (4.24) and (4.25). If we define the following simplifying functions

$$\alpha_\theta = a_1 X^* - 4\mu \sin^2 \frac{\pi\theta}{N}, \quad \beta_\theta = b_2 Y^* - 4\nu \sin^2 \frac{\pi\theta}{N} \quad \text{and} \quad \gamma_\theta = \alpha_\theta + \beta_\theta,$$

and the following constants

$$a = a_2 X^*, \quad b = b_1 Y^*, \quad c_1 = 2b_1 Y^*/NX^* \quad \text{and} \quad c_2 = 2b_2/N,$$

we can write our set of coupled differential equations as

$$\begin{aligned} d\Phi_\theta^X(t)/dt &= 2\alpha_\theta \Phi_\theta^X(t) + 2b\Phi_\theta^{XY}(t) + c_1, \\ d\Phi_\theta^Y(t)/dt &= 2a\Phi_\theta^{XY}(t) + 2\beta_\theta \Phi_\theta^Y(t) + c_2, \quad \text{and} \\ d\Phi_\theta^{XY}(t)/dt &= a\Phi_\theta^X(t) + b\Phi_\theta^Y(t) + \gamma_\theta \Phi_\theta^{XY}(t) \ . \end{aligned} \tag{4.26}$$

4.4 Stationary-State Covariance Analysis

Equations (4.26) represent the inter-relationship between the Fourier transforms of the spatial covariance values as they vary with time. They therefore represent a full *spatial-temporal* covariance result, which we will evaluate later in Section 4.5. First, however, let us look for steady-state (or *stationary*) solutions where we can assume that the time-differentials of all the transform functions ($\Phi_\theta(t)$) are zero. From our deterministic studies (Chapter 3) we know that the linearised system will produce such a stationary state provided the system *instability* is negative (i.e. a sub-critical scenario), and in this case the final state is a uniform stable equilibrium. For the super-critical deterministic case ($I > 0$) no such stationarity exists in the linearised system, since any movement away from initial equilibrium leads to explosive population growth or decay throughout the ring. In comparison, the non-linear system can achieve a stationary state (often with interesting system morphology) in both sub- and super-critical scenarios, and will typically take the general morphological structure of stationary population waves.

4.4.1 Solution of the Simultaneous Equations

If we set all time-differentials to zero in (4.26) we obtain the system of linear simultaneous equations:

$$\begin{aligned} -c_1 &= 2\alpha_\theta \Phi_\theta^X + 2b\Phi_\theta^{XY} \\ -c_2 &= 2a\Phi_\theta^{XY} + 2\beta_\theta \Phi_\theta^Y \\ 0 &= a\Phi_\theta^X + b\Phi_\theta^Y + \gamma_\theta \Phi_\theta^{XY} \ . \end{aligned} \tag{4.27}$$

These equations are easily solved to produce the following expressions for the Fourier transform functions:

$$\Phi_\theta^X = \frac{b^2 c_2 + (\beta_\theta \gamma_\theta - ab)c_1}{2\gamma_\theta(ab - \alpha_\theta\beta_\theta)} \tag{4.28}$$

$$\Phi_\theta^Y = \frac{a^2 c_1 + (\alpha_\theta \gamma_\theta - ab)c_2}{2\gamma_\theta(ab - \alpha_\theta\beta_\theta)} \tag{4.29}$$

$$\Phi_\theta^{XY} = \frac{-ac_1\beta_\theta - bc_2\alpha_\theta}{\gamma_\theta} \; . \tag{4.30}$$

We can then use the inverse transform expressions (4.19) and (4.21) to obtain expressions for the three auto- and cross-covariance functions for cell lag u in the stationary case. These are

$$\begin{aligned}
\sigma_u^X &= \sum_{\theta=0}^{N-1} \frac{b^2 c_2 + (\beta_\theta \gamma_\theta - ab)c_1}{2\gamma_\theta(ab - \alpha_\theta\beta_\theta)} e^{-2\pi i u\theta/N} \\
\sigma_u^Y &= \sum_{\theta=0}^{N-1} \frac{a^2 c_1 + (\alpha_\theta \gamma_\theta - ab)c_2}{2\gamma_\theta(ab - \alpha_\theta\beta_\theta)} e^{-2\pi i u\theta/N} \\
\sigma_u^{XY} &= \sum_{\theta=0}^{N-1} \frac{-ac_1\beta_\theta - bc_2\alpha_\theta}{\gamma_\theta} e^{-2\pi i u\theta/N} \; .
\end{aligned}$$

4.4.2 Covariances for Stationary Sub-Critical Systems

It is a straightforward process to take the exact expressions above for the stationary covariance functions, and evaluate them for a given set of system parameters. In doing this we can obtain a qualitative graphical representation of the behaviour of the covariances for different types of spatial reaction system, and also obtain some quantitative measure of the population structure within a stationary stochastic system. In almost all the numerical evaluations in the following sections, we deal with our generic two-species system, using Turing's *instability* parameter (I) as the main tool for investigating our parameter space. We therefore choose to set the following parameter values (as used for the standard realisations detailed in Chapter 3): $a_1 = (I-2)/X^*, b_1 = -1.25/X^*, a_2 = 2.5/Y^*$ and $b_2 = (I + 1.5)/Y^*$, with $X^* = Y^* = 10$, on a ring of $N = 50$ cells with the migration rates set to produce five *waves* with a migration rate ratio $m = \mu/\nu = 2$. By choosing these parameters we can use the sign of I to control the criticality of the system — positive I-values give super-critical behaviour, whereas negative values produce the sub-critical system.

The sub-critical system is the most obvious for us to investigate as it is in this scenario that we observe stationary behaviour in our linearised deterministic studies. In the linearised scenario any perturbation to the equilibrium state of the super-critical deterministic system will result in exponential explosion of the cell population, and hence no stationary state. We will see later in Section 4.5.5 that the stochastic spatial-temporal covariances undergo similar explosions as the super-critical system evolves in time. We can therefore be certain that any numerical calculation of stationary state values must be flawed.

Figure 4.1 shows the numerical evaluation of the three covariance functions when the system is only just in the sub-critical domain — with $I = -0.001$. We observe a very

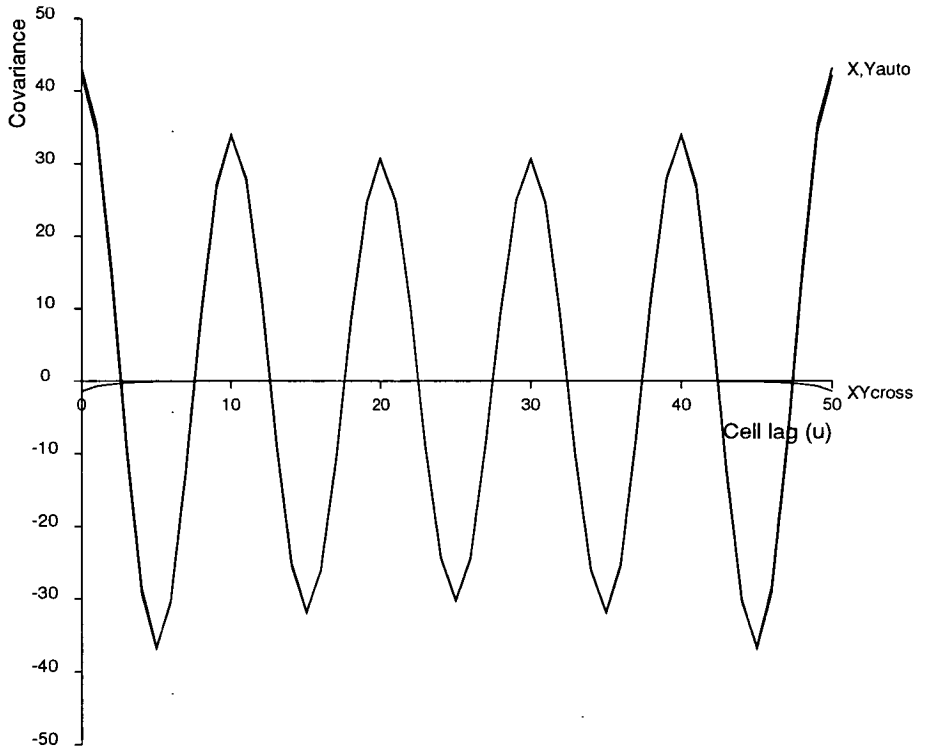

Figure 4.1: Low interaction stationary-state covariance structure
Using our standard system parameters with a very small negative instability ($I = -0.001$), we observe a very strong wave structure in both the X and Y auto-covariance values. The XY cross-covariance measure is negligible in comparison, and shows no interesting spatial structure, other than a slight local negative correlation.

strong spatial wave pattern in the covariance values, with the wavelength agreeing with that expected from our parameter set (i.e. we obtain five waves). It is interesting to note the almost identical behaviour of both X and Y auto-covariances, and the negligible effect of the XY cross-covariance at all cell lags other than $u = 0$. The graph in Figure 4.1 indicates that for a linear stochastic spatial reaction system that lies just

inside the sub-critical domain of parameter space, there will be a strong correlation between the perturbations that occur (due to the Gaussian noise structure added to the deterministic model) at cells that are multiples of ten cells apart. Additionally, there is a strong negative correlation between those population perturbations that are separated by $10p + 5$ cells, where $p =$ is any integer. It is also worthy of note that although the correlation reduces slightly (by approximately 20%) as the cell lag moves to the furthest point away on the ring, the auto-correlations have a strong influence throughout the whole system.

Figure 4.2 shows the effects of making two slight adjustments to our system parameters. The first is to increase the magnitude of the system instability (still maintaining a sub-critical scenario), and the second is to adapt the migration rates in the system, in order to produce parameters that create ten population waves in the deterministic case.

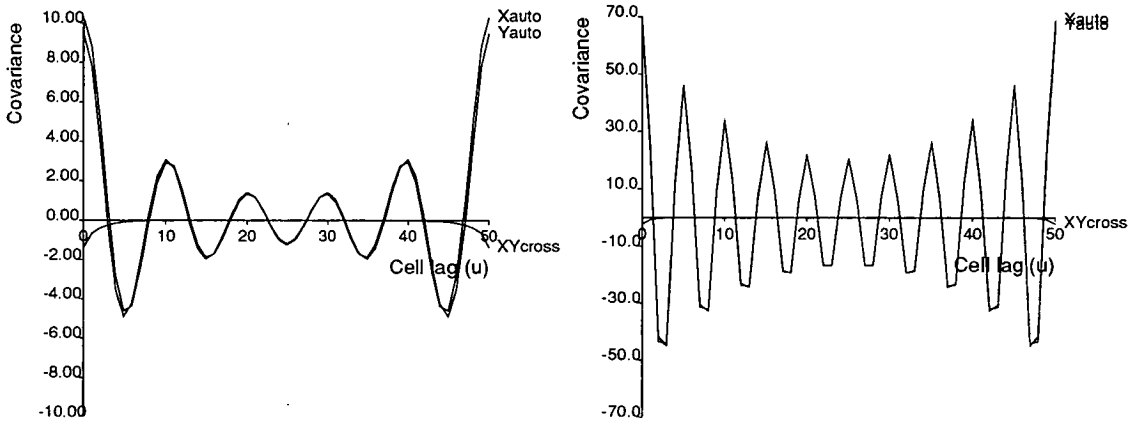

Figure 4.2: Medium interaction stationary-state covariance structures
Again using our standard parameter set, these graphs show the effects of increasing the magnitude of the sub-critical instability (left-hand graph, $I = -0.01$), and increasing migration rates to produce ten deterministic waves (right-hand graph, $I = -0.001$).

In both these new scenarios we see that the strength of the auto-covariance for both X and Y is substantially reduced (by approximately 80%) as the cell lag approaches the furthest point away on the ring. However, the wave structure is still very evident, and so we can expect to see a regular wave pattern in the stochastic populations using these parameters. Note that for the case of increased migration rates, the deterministic wavelength has been exactly preserved for the stochastic auto-covariances. In addition, the cross-covariance values again appear as negligible, except for their slight negative correlation for very small cell lag values ($u < \pm 5$).

Figure 4.3 shows that using a relatively high magnitude of negative instability value

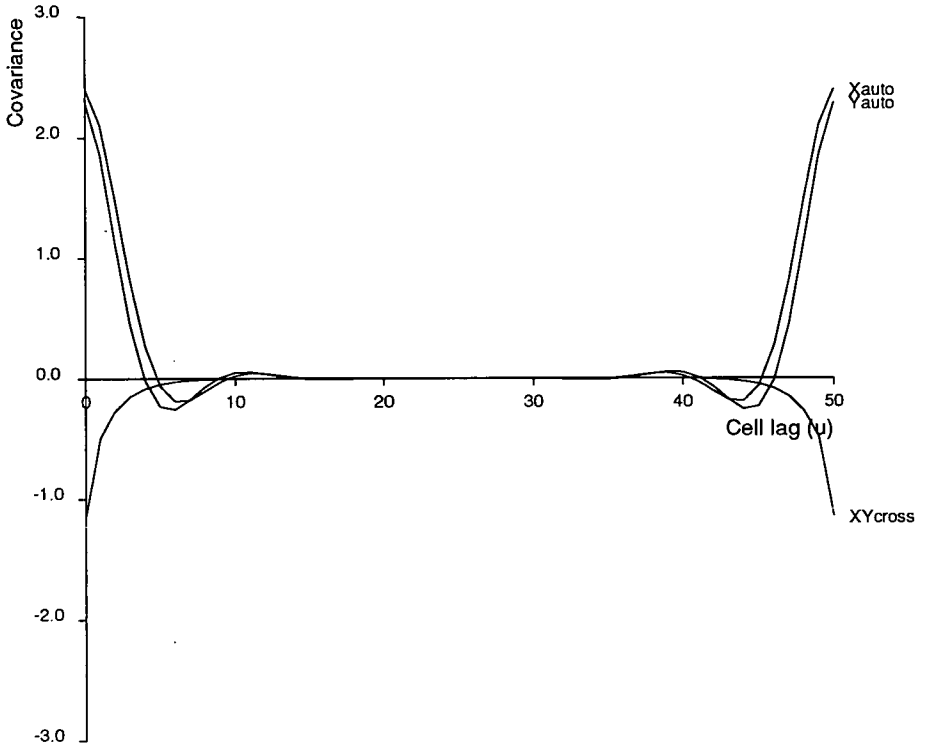

Figure 4.3: High interaction stationary-state covariance structure
Exact auto- and cross-covariances for perturbations in our standard sub-critical system with high magnitude instability ($I = -0.1$). All three covariance functions are only non-negligible for small cell lag values. This highlights that only short range correlation effects are produced as the sub-critical system instability magnitude increases.

such as $I = -0.1$ has a dramatic effect on the covariance functions within our stochastic systems. We now see that all positive correlation is restricted to small cell lag values. This knowledge of the correlation-length between perturbations is of great use for implementing efficient parallel versions of such stochastic simulations, and we will cover such issues in later chapters. The details from Figure 4.3 show that there is a significant positive auto-correlation for both X- and Y-perturbations only when $u \leq 2$; for $u = 3$ and $10 \leq u \leq 13$ the correlation is very small but still positive; and for $4 \leq u \leq 9$ we have a small negative correlation. For all larger values of u we observe that auto- and cross-covariance values are negligible. This pattern suggests a system where all perturbations in either population will have a strong short-range effect, creating reinforcement of the perturbation in local and near-neighbour cells, plus a very small effect on those cells approximately one wavelength distant (in this case ten cells). In addition there will be a small inhibitory effect on perturbations in cells a half-wavelength distant from the initial perturbation. It is clear that such a system can contain many

independent concentrations of reactants (localised "waves" or *hot-spots*) that will have little effect on each other apart from a tendency to inhibit similar perturbations at a half-wavelength distance. This is very different from the small magnitude instabilities that produce such strong long-range influences.

This trend continues as we again increase the magnitude of our sub-critical instability: with $I = -1.0$ numerical evaluation of the exact covariance functions produces values that are small for almost all cell lag values, e.g. the X- and Y-auto-covariances are both close to 0.07 for $u = 0$, and equal 0.03 when $u = 1$. We thus see that as the sub-critical system instability decreases from zero, there is a marked reduction in the magnitude of correlations between perturbations. This analytic result can be compared directly to the stochastic simulation results in Chapter 5.

4.5 Spatial-Temporal Covariance Analysis

Following success with the solution of the covariance equations at stationarity, let us now solve the system in the general time-dependent case. This will generate expressions for spatial covariances as they changes in time, thereby giving us a very powerful to tool to examine and predict the behaviour of our general stochastic spatial reaction systems.

Let us therefore re-examine the coupled differential equations (4.26) for the Fourier transforms of our second-order moments. Such a system can either be solved using standard simultaneous differential equation techniques, or by treating it as a vector differential equation with a matrix of coefficients. These approaches are equivalent, and both ultimately result in the calculation of the determinants of the coefficient matrix. Let us tackle the problem from the former approach in order to concentrate specifically on the two auto-covariance values. Our coupled equations are

$$d\Phi_\theta^X(t)/dt = 2\alpha_\theta \Phi_\theta^X(t) + 2b\Phi_\theta^{XY}(t) + c_1 \tag{4.31}$$

$$d\Phi_\theta^Y(t)/dt = 2a\Phi_\theta^{XY}(t) + 2\beta_\theta \Phi_\theta^Y(t) + c_2 \tag{4.32}$$

$$d\Phi_\theta^{XY}(t)/dt = a\Phi_\theta^X(t) + b\Phi_\theta^Y(t) + \gamma_\theta \Phi_\theta^{XY}(t) \ . \tag{4.33}$$

We can reorganise (4.31) to give

$$\Phi_\theta^{XY}(t) = (1/2b)(d\Phi_\theta^X(t)/dt - 2\alpha_\theta\Phi_\theta^X(t) - c_1), \tag{4.34}$$

and (4.32) to produce

$$\Phi_\theta^{XY}(t) = (1/2a)(d\Phi_\theta^Y(t)/dt - 2\beta_\theta\Phi_\theta^Y(t) - c_2) \ . \tag{4.35}$$

If we now substitute (4.35) into (4.33) we obtain

$$\frac{d}{dt}\left[\frac{1}{2a}\left(\frac{d\Phi_\theta^Y(t)}{dt} - 2\beta_\theta\Phi_\theta^Y(t) - c_2\right)\right]$$

$$= a\Phi_\theta^X(t) + b\Phi_\theta^Y(t) + \frac{\gamma_\theta}{2a}\left[\frac{d\Phi_\theta^Y(t)}{dt} - 2\beta_\theta\Phi_\theta^Y(t) - c_2\right] \ .$$

A little rearrangement gives us the following second-order differential equation

$$\Phi_\theta^X(t) = \frac{1}{2a^2}\left[\frac{d^2\Phi_\theta^Y(t)}{dt^2} - (\alpha_\theta + 3\beta_\theta)\frac{d\Phi_\theta^Y(t)}{dt} + 2(\beta_\theta\gamma_\theta - ab)\Phi_\theta^Y(t) + c_2\gamma_\theta\right] \ . \tag{4.36}$$

If we now substitute (4.36) and (4.35) into Equation (4.31) we obtain a third-order differential equation in $\Phi_\theta^Y(t)$. This reduces to

$$d^3\Phi_\theta^Y(t)/dt^3 - 3\gamma_\theta d^2\Phi_\theta^Y(t)/dt^2 + 2(\gamma_\theta^2 + 2(\alpha_\theta\beta_\theta - ab))d\Phi_\theta^Y(t)/dt$$
$$+4\gamma_\theta(ab - \alpha_\theta\beta_\theta)\Phi_\theta^Y(t) = 2(a^2c_1 + (\alpha_\theta\gamma_\theta - ab)c_2) \ . \tag{4.37}$$

Following a similar procedure we can also obtain a third-order equation for the X covariance. Substituting (4.34) into (4.33) gives us

$$\Phi_\theta^Y(t) = \frac{1}{2b^2}\left[\frac{d^2\Phi_\theta^X(t)}{dt^2} - (3\alpha_\theta + \beta_\theta)\frac{d\Phi_\theta^X(t)}{dt} + 2(\alpha_\theta\gamma_\theta - ab)\Phi_\theta^X(t) + c_1\gamma_\theta\right], \tag{4.38}$$

which can in turn be substituted into (4.32), with (4.34), to produce

$$d^3\Phi_\theta^X(t)/dt^3 - 3\gamma_\theta d^2\Phi_\theta^X(t)/dt^2 + 2(\gamma_\theta^2 + 2(\alpha_\theta\beta_\theta - ab))d\Phi_\theta^X(t)/dt$$
$$+4\gamma_\theta(ab - \alpha_\theta\beta_\theta)\Phi_\theta^X(t) = 2(b^2c_2 + (\beta_\theta\gamma_\theta - ab)c_1) \ . \tag{4.39}$$

4.5.1 Solution of Third-Order ODEs

We now have two almost identical third-order inhomogeneous ordinary differential equations for the Fourier transform of the X and Y perturbation auto-covariances. We must find complete solutions of these equations in order to determine the nature of the full spatial-temporal auto-covariance structure. This task is assisted by noting that the two ordinary differential equations (4.37) and (4.39) are identical in their homogeneous form; it is only the constant inhomogeneous term on the right-hand side of the equations that distinguishes them. This allows us to find just one general solution for the homogeneous case of both equations, to which we can add different particular solutions to complete the full inhomogeneous solution.

Thus, let us put (4.37) and (4.39) into the general form

$$\frac{d^3\Phi_\theta^s(t)}{dt^3} + 3\mathcal{B}_\theta\frac{d^2\Phi_\theta^s(t)}{dt^2} + 3\mathcal{C}_\theta\frac{d\Phi_\theta^s(t)}{dt} + \mathcal{D}_\theta\Phi_\theta^s(t) = C_s, \tag{4.40}$$

where s indexes the two reactant types X and Y. Thus

$$C_X = 2(b^2c_2 + c_1(\beta_\theta\gamma_\theta - ab)) \quad \text{and} \quad C_Y = 2(a^2c_1 + c_2(\alpha_\theta\gamma_\theta - ab)), \tag{4.41}$$

and the equation coefficients are

$$\mathcal{B}_\theta = -\gamma_\theta\ , \quad \mathcal{C}_\theta = 2(\gamma_\theta^2 + 2(\alpha_\theta\beta_\theta - ab))/3 \quad \text{and} \quad \mathcal{D}_\theta = 4\gamma_\theta(ab - \alpha_\theta\beta_\theta)\ . \tag{4.42}$$

Solution of the homogeneous case

Taking the homogeneous case of (4.40), we can follow standard theory (see Sneddon [1976]) to write the auxiliary equation for the system as the cubic

$$\phi^3 + 3\mathcal{B}_\theta\phi^2 + 3\mathcal{C}_\theta\phi + \mathcal{D}_\theta = 0\ . \tag{4.43}$$

In the general case this equation will have three roots, ϕ_j $(j = 1, 2, 3)$, one of which must always be real, but two of which could be complex conjugates, say $\phi_{2,3} = \varphi \pm i\varpi$. This result is dependent upon the *discriminant* (Δ) condition that we will examine later.

The general homogeneous solution will therefore take the form

$$\Phi_\theta(t) = A_1 e^{\phi_1 t} + A_2 e^{\phi_2 t} + A_3 e^{\phi_3 t}$$

if all ϕ_i are real, and

$$\Phi_\theta(t) = B_1 e^{\phi_1 t} + B_2 e^{\varphi t} \cos \varpi t + B_3 e^{\varphi t} \sin \varpi t,$$

if two of the roots are complex.

Again using standard theory, if we introduce a new variable (ψ) in (4.43), so that $\phi = \psi - \mathcal{B}_\theta$, we can rewrite the auxiliary equation as

$$\psi^3 + 3p\psi + q = 0 \tag{4.44}$$

where

$$p = \mathcal{C}_\theta - \mathcal{B}_\theta^2 \quad \text{and} \quad q = \mathcal{D}_\theta - 3\mathcal{B}_\theta\mathcal{C}_\theta + 2\mathcal{B}_\theta^3 \ .$$

In this form we can write the equation discriminant as

$$\Delta = 4p^3 + q^2 \ .$$

Cubic equation theory states that if $\Delta < 0$ the equation will have three distinct real roots, if $\Delta = 0$ we obtain 3 real roots, two of which are equal (except in the special case of $p = q = 0$ when all three are equal), and if $\Delta > 0$ we get one real and two conjugate complex roots. Using (4.42) we can simplify this discriminant since

$$q^2 = (\mathcal{D}_\theta - 3\mathcal{B}_\theta\mathcal{C}_\theta + 2\mathcal{B}_\theta^3)^2 = 0 \quad \text{and} \quad p^3 = (\mathcal{C}_\theta - \mathcal{B}_\theta^2)^3 = -[(\alpha_\theta - \beta_\theta)^2 + 4ab]^3/27 \ .$$

Let us define $\omega_\theta^2 = (\alpha_\theta - \beta_\theta)^2 + 4ab = -3p$, since we see that the nature of our homogeneous solution is governed by the sign of ω_θ^2. If $\omega_\theta^2 > 0$ then $\Delta < 0$ and we have three real roots, whereas if $\omega_\theta^2 < 0$ we get $\Delta > 0$ and a complex solution.

Incorporating $q = 0$ into (4.44) we can write the solutions as $\psi = 0, \sqrt{-3p}, -\sqrt{-3p}$. This allows us to revert back to our original variable ϕ to obtain the solutions $\phi_j = -\mathcal{B}_\theta, \omega_\theta - \mathcal{B}_\theta, -\omega_\theta - \mathcal{B}_\theta$, i.e. $\phi_j = \gamma_\theta, \gamma_\theta \pm \omega_\theta$. We can therefore write the general solutions to the homogeneous equations for the Fourier transforms of both the X and Y

auto-covariances as

$$\Phi_\theta^s(t) = e^{\gamma_\theta t}(A_1 + A_2 e^{\omega_\theta t} + A_3 e^{-\omega_\theta t}) \text{ if } \omega_\theta^2 \geq 0 \text{ and} \tag{4.45}$$

$$\Phi_\theta^s(t) = e^{\gamma_\theta t}(B_1 + B_2 \cos\omega_\theta t + B_3 \sin\omega_\theta t) \text{ if } \omega_\theta^2 < 0, \tag{4.46}$$

where s indexes the two reactant types X and Y. The coefficients A_j and B_j are dependent upon initial conditions, as well as the particular solution (C_s), the latter must be added to give the full inhomogeneous solution.

Solution of the general inhomogeneous case

In order to obtain a complete solution to the full inhomogeneous third-order ODEs for $\Phi_\theta^X(t)$ and $\Phi_\theta^Y(t)$ (4.40) we must add a particular solution ($\Psi^s(t)$) to the homogeneous result ($\Phi_\theta^s(t)$) which is given above in Equations (4.45) and (4.46). If we consider the technique of *variation of parameters* (see Greenspan [1960]) we can detail the general methodology for all cases (X and Y, and $\omega_\theta^2 \geq 0$ and $\omega_\theta^2 < 0$), before providing precise results for each.

Considering the general particular solution $\Psi(t)$ as a summation of terms linear in each of the functions making the general homogeneous solution ($\phi_j(t)$ say), but with potentially time-dependent coefficients $k_j(t)$, we can write

$$\Psi(t) = k_1(t)\phi_1(t) + k_2(t)\phi_2(t) + k_3(t)\phi_3(t), \tag{4.47}$$

where, for example, if $\omega_\theta^2 > 0$ then $\phi_1(t) = e^{\gamma_\theta t}$ and $\phi_2(t) = e^{(\gamma_\theta + \omega_\theta^2)t}$ etc.

Using the variation of parameters technique we now perform a succession of differentiations with respect to time. For simplicity and clarity we specify this by ($'$), and remove all time dependent references from the equations. Thus, differentiation of (4.47) produces

$$\Psi' = k_1'\phi_1 + k_2'\phi_2 + k_3'\phi_3 + k_1\phi_1' + k_2\phi_2' + k_3\phi_3' \ .$$

The standard approach assumes that we can set

$$k_1'\phi_1 + k_2'\phi_2 + k_3'\phi_3 = 0 \ . \tag{4.48}$$

Differentiating again with respect to time gives

$$\Psi'' = k_1'\phi_1' + k_2'\phi_2' + k_3'\phi_3' + k_1\phi_1'' + k_2\phi_2'' + k_3\phi_3'' \ .$$

With a similar assumption that

$$k_1'\phi_1' + k_2'\phi_2' + k_3'\phi_3' = 0, \tag{4.49}$$

we can perform a final differentiation to obtain

$$\Psi''' = k_1'\phi_1'' + k_2'\phi_2'' + k_3'\phi_3'' + k_1\phi_1''' + k_2\phi_2''' + k_3\phi_3''' \ .$$

If we now reconsider the original inhomogeneous equation (4.40), and substitute our particular solution Ψ into this equation, this gives

$$\Psi''' + 3\mathcal{B}_\theta\Psi'' + 3\mathcal{C}_\theta\Psi' + \mathcal{D}_\theta\Psi = C_s \ .$$

Substituting the above expressions into this equation produces

$$k_1'\phi_1'' + k_2'\phi_2'' + k_3'\phi_3'' + k_j(\phi_j''' + 3\mathcal{B}_\theta\phi_j'' + 3\mathcal{C}_\theta\phi_j' + \mathcal{D}_\theta\phi_j) = C_s,$$

where $j = 1, 2, 3$. Since the ϕ_j are solutions of the homogeneous version of (4.40) we can write

$$\phi_j''' + 3\mathcal{B}_\theta\phi_j'' + 3\mathcal{C}_\theta\phi_j' + \mathcal{D}_\theta\phi_j = 0,$$

and thus

$$k_1'\phi_1'' + k_2'\phi_2'' + k_3'\phi_3'' = C_s \ . \tag{4.50}$$

Therefore equations (4.48), (4.49) and (4.50) form a set of simultaneous linear equations for the first temporal differential of the coefficients, $k_j(t)$, in the particular solution. Given known functions $\phi_j(t)$ and their first and second differentials with respect to time, we can solve these systems exactly using standard linear algebra techniques. Thus, as detailed in Sneddon [1976],

$$k_1' = C_s\phi_1'\frac{\phi_3'\phi_2 - \phi_2'\phi_3}{F(\phi_j)} \ , \quad k_2' = C_s\phi_1'\frac{\phi_3\phi_1' - \phi_1\phi_3'}{F(\phi_j)} \quad \text{and} \quad k_3' = C_s\phi_1'\frac{\phi_1'\phi_2 - \phi_2'\phi_1}{-F(\phi_j)}, \tag{4.51}$$

where

$$F(\phi_j) = (\phi_3'\phi_1'' - \phi_3''\phi_1')(\phi_2\phi_1' - \phi_2'\phi_1) - (\phi_3\phi_1' - \phi_3'\phi_1)(\phi_2'\phi_1'' - \phi_1'\phi_2'') \ . \quad (4.52)$$

4.5.2 Exact Solution of the Real System

If we concentrate specifically on the solution of the full (inhomogeneous) ordinary differential equation (4.40) when we have three real roots for the auxiliary equation (i.e. $\omega_\theta^2 > 0$), we then have a particular solution in the form

$$\Psi^s(t) = k_1(t)e^{\gamma_\theta t} + k_2(t)e^{(\gamma_\theta+\omega_\theta)t} + k_3(t)e^{(\gamma_\theta-\omega_\theta)t}, \quad (4.53)$$

(where again s denotes the reactant type X or Y) and thus we have the general homogeneous solution functions

$$\phi_1(t) = e^{\gamma_\theta t} \ , \quad \phi_2(t) = e^{(\gamma_\theta+\omega_\theta)t} \quad \text{and} \quad \phi_3(t) = e^{(\gamma_\theta-\omega_\theta)t} \ .$$

These functions all have straightforward first and second differentials with respect to time. Thus substitution of these functions and their differentials into (4.52) gives

$$F(\phi_j) = 2\omega_\theta^3\gamma_\theta e^{4\gamma_\theta t} \ .$$

This allows us to specify the time-dependent coefficient functions as

$$k_1'(t) = -\frac{C_s e^{-\gamma_\theta t}}{\omega_\theta^2} \ , \quad k_2'(t) = \frac{C_s e^{-(\gamma_\theta+\omega_\theta)t}}{2\omega_\theta^2} \quad \text{and} \quad k_3'(t) = \frac{C_s e^{-(\gamma_\theta-\omega_\theta)t}}{2\omega_\theta^2} \ . \quad (4.54)$$

We can integrate all of the expressions in (4.54), and since we are looking for any particular solution to this system, we can freely ignore any constants of integration. This provides us with

$$k_1(t) = \frac{C_s e^{-\gamma_\theta t}}{\omega_\theta^2\gamma_\theta} \ , \quad k_2(t) = -\frac{C_s e^{-(\gamma_\theta+\omega_\theta)t}}{2\omega_\theta^2(\gamma_\theta+\omega_\theta)} \quad \text{and} \quad k_3(t) = -\frac{C_s e^{-(\gamma_\theta-\omega_\theta)t}}{2\omega_\theta^2(\gamma_\theta-\omega_\theta)} \ . \quad (4.55)$$

Substituting the coefficient functions (4.55) into the particular (real) solution (4.53) we find that all the time-dependent terms cancel and we obtain a constant particular

solution. We thus obtain the real case particular solution:

$$\Psi^s = -C_s/\gamma_\theta(\gamma_\theta^2 - \omega_\theta^2) = -C_s/4\gamma_\theta(\alpha_\theta\beta_\theta - ab), \tag{4.56}$$

which therefore allows us to write the general solution to the inhomogeneous equations for the auto-covariance transforms (with $\omega_\theta^2 \geq 0$) as

$$\Phi_\theta^s(t) = e^{\gamma_\theta t}(A_1 + A_2 e^{\omega_\theta t} + A_3 e^{-\omega_\theta t}) - C_s/\gamma_\theta(\gamma_\theta^2 - \omega_\theta^2) \ . \tag{4.57}$$

In order to determine the coefficients (A_j) of this general solution, we use our knowledge of the initial system conditions. Assuming our system has no initial spatial structure, with all perturbation values $x_r(0) = y_r(0) = 0$ for $r = 1, \ldots, N$, we see that all initial second-order moments (4.14) equate to zero at $t = 0$. Thus we also find from (4.18) and (4.20) that the values of the Fourier transforms of these second-order moments also equal zero, i.e. $\Phi_\theta^X(0) = \Phi_\theta^Y(0) = \Phi_\theta^{XY}(0) = 0$. In addition we can determine the value of the first and second differentials of the Fourier transforms at $t = 0$. Studying (4.31), (4.32) and (4.33) reveals that

$$d\Phi_\theta^X(0)/dt = c_1 \ , \ \ d\Phi_\theta^Y(0)/dt = c_2 \ \text{ and } \ d\Phi_\theta^{XY}(0)/dt = 0, \tag{4.58}$$

and differentiation of (4.31) and (4.32) gives

$$d^2\Phi_\theta^X(0)/dt^2 = 2\alpha_\theta d\Phi_\theta^X(0)/dt + 2bd\Phi_\theta^{XY}(0)/dt = 2\alpha_\theta c_1 \ \text{ and}$$

$$d^2\Phi_\theta^Y(0)/dt^2 = 2ad\Phi_\theta^{XY}(0)/dt + 2\beta_\theta\Phi_\theta^Y(0)/dt = 2\beta_\theta c_2 \ . \tag{4.59}$$

We can now differentiate (4.57) twice to give alternative expressions for the coefficients A_j as

$$d\Phi_\theta^s(t)/dt = e^{\gamma_\theta t}(\gamma_\theta A_1 + (\gamma_\theta + \omega_\theta)A_2 e^{\omega_\theta t} + (\gamma_\theta - \omega_\theta)A_3 e^{-\omega_\theta t}) \ \text{ and} \tag{4.60}$$

$$d^2\Phi_\theta^s(t)/dt^2 = e^{\gamma_\theta t}(\gamma_\theta^2 A_1 + (\gamma_\theta + \omega_\theta)^2 A_2 e^{\omega_\theta t} + (\gamma_\theta - \omega_\theta)^2 A_3 e^{-\omega_\theta t}) \ . \tag{4.61}$$

Taking (4.57), (4.60) and (4.61) at $t = 0$ gives us the set of linear simultaneous equations

$$A_1 + A_2 + A_3 = C_s/\gamma_\theta(\gamma_\theta^2 - \omega_\theta^2),$$

$$A_1 + (\gamma_\theta + \omega_\theta)A_2 + (\gamma_\theta - \omega_\theta)A_3 = c_s, \quad \text{and} \tag{4.62}$$

$$\gamma_\theta^2 A_1 + (\gamma_\theta + \omega_\theta)^2 A_2 + (\gamma_\theta - \omega_\theta)^2 A_3 = 2p_s c_s \ .$$

Here C_s is defined by (4.41), $c_s = c_1$ for the X-reactant and $c_s = c_2$ for Y, and $p_s = \alpha_\theta$ for the X-reactant and $p_s = \beta_\theta$ for Y. Equations (4.62) solve to give

$$\begin{aligned}
A_1 &= \frac{-C_s}{\gamma_\theta \omega_\theta^2}\left[1 - \frac{2\gamma_\theta c_s}{C_s}(\gamma_\theta - p_s)\right] \\
A_2 &= \frac{C_s}{2\omega_\theta^2(\gamma_\theta + \omega_\theta)}\left[1 - \frac{(\gamma_\theta + \omega_\theta)c_s}{C_s}(2(\gamma_\theta - p_s) - \omega_\theta)\right] \quad \text{and} \\
A_3 &= \frac{C_s}{2\omega_\theta^2(\gamma_\theta - \omega_\theta)}\left[1 - \frac{(\gamma_\theta - \omega_\theta)c_s}{C_s}(2(\gamma_\theta - p_s) + \omega_\theta)\right] \ .
\end{aligned}$$

These values can now be inserted into (4.57) to obtain an exact expression for the Fourier transform of the spatial-temporal auto-covariance values in the real domain, namely

$$\begin{aligned}
\Phi_\theta^s(t) = \Bigg(& \gamma_\theta(\gamma_\theta - \omega_\theta)e^{(\gamma_\theta + \omega_\theta)t}\left[1 - \frac{(\gamma_\theta + \omega_\theta)c_s}{C_s}(2(\gamma_\theta - p_s) - \omega_\theta)\right] \\
& + \gamma_\theta(\gamma_\theta + \omega_\theta)e^{(\gamma_\theta - \omega_\theta)t}\left[1 - \frac{(\gamma_\theta - \omega_\theta)c_s}{C_s}(2(\gamma_\theta - p_s) + \omega_\theta)\right] \\
& - 2(\gamma_\theta^2 - \omega_\theta^2)e^{\gamma_\theta t}\left[1 - \frac{2\gamma_\theta c_s}{C_s}(\gamma_\theta - p_s)\right] - 2\omega_\theta^2\Bigg)\frac{C_s}{2\gamma_\theta\omega_\theta^2(\gamma_\theta^2 - \omega_\theta^2)} \ .
\end{aligned} \tag{4.63}$$

4.5.3 Exact Solution of the Complex System

Let us now consider the scenario where we have two complex roots for the auxiliary equation (4.43), i.e. we have $\omega_\theta^2 < 0$. We now have a solution of the form shown in (4.46), but with the addition of a particular solution ($\hat{\Psi}^s(t)$). If we again use the variation of parameters technique detailed in Section 4.5.1 we look for a particular solution of the form

$$\hat{\Psi}^s(t) = l_1(t)e^{\gamma_\theta t} + l_2(t)e^{\gamma_\theta t}\cos\omega_\theta t + l_3(t)e^{\gamma_\theta t}\sin\omega_\theta t \ . \tag{4.64}$$

Thus, as in Section 4.5.2 we can obtain the coefficient functions $l_j(t)$ through a succession of linear manipulations. We obtain

$$l_1(t) = -\frac{C_s e^{-\gamma_\theta t}}{\omega_\theta^2\gamma_\theta} \ , \quad l_2(t) = \frac{C_s e^{-\gamma_\theta t}(\gamma_\theta \cos\omega_\theta t - \omega_\theta \sin\omega_\theta t)}{\omega_\theta^2(\gamma_\theta^2 + \omega_\theta^2)} \quad \text{and}$$

$$l_3(t) = \frac{C_s e^{-\gamma_\theta t}(\gamma_\theta \sin \omega_\theta t + \omega_\theta \cos \omega_\theta t)}{\omega_\theta^2(\gamma_\theta^2 + \omega_\theta^2)},$$

which produce the constant particular solution

$$\hat{\Psi}^s(t) = -C_s/\gamma_\theta(\gamma_\theta^2 + \omega_\theta^2) = -C_s/2\gamma_\theta(\alpha_\theta^2 + \beta_\theta^2 + 2ab) \ . \tag{4.65}$$

We can therefore write the general solution to the inhomogeneous equations for the auto-covariance transforms (with $\omega_\theta^2 < 0$) as

$$\Phi_\theta^s(t) = e^{\gamma_\theta t}(B_1 + B_2 \cos \omega_\theta t + B_3 \sin \omega_\theta t) - C_s/\gamma_\theta(\gamma_\theta^2 + \omega_\theta^2) \ . \tag{4.66}$$

Using the initial conditions as specified in Section 4.5.2, we can find values for the constants B_j in (4.66) and in its first and second derivatives. This provides us with another set of three linear equations. These can be solved straightforwardly to produce the coefficient values

$$B_1 = \frac{C_s}{\gamma_\theta \omega_\theta^2}\left[1 - \frac{2\gamma_\theta c_s}{C_s}(\gamma_\theta - p_s)\right], B_2 = \frac{-\gamma_\theta C_s}{\omega_\theta^2(\gamma_\theta^2 + \omega_\theta^2)}\left[1 - \frac{2(\gamma_\theta^2 + \omega_\theta^2)c_s}{\gamma_\theta C_s}(\gamma_\theta - p_s)\right]$$

$$\text{and} \quad B_3 = \frac{-C_s}{\omega_\theta(\gamma_\theta^2 + \omega_\theta^2)}\left[1 - \frac{(\gamma_\theta^2 + \omega_\theta^2)c_s}{C_s}\right] \ .$$

These values can now be inserted into (4.66) to obtain an exact expression for the Fourier transform of the spatial-temporal auto-covariances for reactant perturbation values in the complex domain, thus

$$\begin{aligned} \Phi_\theta^s(t) = \frac{C_s}{\gamma_\theta \omega_\theta^2(\gamma_\theta^2 + \omega_\theta^2)}\Bigg(& - \gamma_\theta^2 \cos \omega_\theta t e^{\gamma_\theta t}\left[1 - \frac{2(\gamma_\theta^2 + \omega_\theta^2)c_s}{\gamma_\theta C_s}(\gamma_\theta - p_s)\right] \\ & - \gamma_\theta \omega_\theta \sin \omega_\theta t e^{\gamma_\theta t}\left[1 - \frac{(\gamma_\theta^2 + \omega_\theta^2)c_s}{C_s}\right] \\ & + (\gamma_\theta^2 + \omega_\theta^2)e^{\gamma_\theta t}\left[1 - \frac{2\gamma_\theta c_s}{C_s}(\gamma_\theta - p_s)\right] - \omega_\theta^2\Bigg) \ . \end{aligned} \tag{4.67}$$

4.5.4 Unified Spatial-Temporal Auto-Covariances

Studying Equation (4.67), we notice that in this *complex* scenario ($\omega_\theta^2 < 0$) there are a number complex terms (i.e. terms in single powers of ω_θ). Ideally we would like to

remove such terms, since this would allow us to unify our auto-covariance solutions for both real and complex systems. Thus, for the $\omega_\theta^2 < 0$ scenario, let us define

$$\omega_\theta = i\sqrt{|\omega_\theta^2|} = i\omega'_\theta \text{ and } \omega'_\theta = \sqrt{-\omega_\theta^2} \ .$$

This allows us to write (4.67) as

$$\begin{aligned}\Phi_\theta^s(t) = \Bigg(&-\gamma_\theta^2 \cosh(\omega'_\theta t)e^{\gamma_\theta t}\left[1 - \frac{2(\gamma_\theta^2 - (\omega'_\theta)^2)c_s}{\gamma_\theta C_s}(\gamma_\theta - p_s)\right] \\ &+\gamma_\theta\omega'_\theta \sinh(\omega'_\theta t)e^{\gamma_\theta t}\left[1 - \frac{(\gamma_\theta^2 - (\omega'_\theta)^2)c_s}{C_s}\right] \\ &+ (\gamma_\theta^2 - (\omega'_\theta)^2)e^{\gamma_\theta t}\left[1 - \frac{2\gamma_\theta c_s}{C_s}(\gamma_\theta - p_s)\right] + (\omega'_\theta)^2\Bigg)\frac{-C_s}{\gamma_\theta(\omega'_\theta)^2(\gamma_\theta^2 - (\omega'_\theta)^2)} \ .\end{aligned}$$

Using the standard relations $\sinh k = (e^k - e^{-k})/2$ and $\cosh k = (e^k + e^{-k})/2$, we can reduce the above equation to

$$\begin{aligned}\Phi_\theta^s(t) = \Bigg(&\gamma_\theta(\gamma_\theta - \omega'_\theta)e^{(\gamma_\theta+\omega'_\theta)t}\left[1 - \frac{(\gamma_\theta + \omega'_\theta)c_s}{C_s}(2(\gamma_\theta - p_s) - \omega'_\theta)\right] \\ &+ \gamma_\theta(\gamma_\theta + \omega'_\theta)e^{(\gamma_\theta-\omega'_\theta)t}\left[1 - \frac{(\gamma_\theta - \omega'_\theta)c_s}{C_s}(2(\gamma_\theta - p_s) + \omega'_\theta)\right] \\ &- 2(\gamma_\theta^2 - (\omega'_\theta)^2)e^{\gamma_\theta t}\left[1 - \frac{2\gamma_\theta c_s}{C_s}(\gamma_\theta - p_s)\right] - 2(\omega'_\theta)^2\Bigg)\frac{C_s}{2\gamma_\theta(\omega'_\theta)^2(\gamma_\theta^2 - (\omega'_\theta)^2)} \ .\end{aligned} \tag{4.68}$$

It can be seen that result (4.68) is identical to the real scenario solution (4.63), but with ω_θ replaced with ω'_θ, where $\omega'_\theta = \omega_\theta$ when $\omega_\theta^2 > 0$ and $\omega'_\theta = \sqrt{-\omega_\theta^2}$ when $\omega_\theta^2 < 0$. We can therefore write down exact (unified) expressions for the Fourier transform representations of the X and Y spatial-temporal auto-covariance functions for our one-dimensional stochastic system. Using (4.41) to define C_s values, (4.58) to give c_s values, and (4.59) for p_s values, we obtain

$$\begin{aligned}\Phi_\theta^X(t) &= C_1\Bigg(\gamma_\theta(\gamma_\theta - \omega'_\theta)e^{(\gamma_\theta+\omega'_\theta)t}\left[1 - \frac{c_1(\gamma_\theta + \omega'_\theta)(2\beta_\theta - \omega'_\theta)}{2(b^2c_2 + c_1(\beta_\theta\gamma_\theta - ab))}\right] \\ &+ \gamma_\theta(\gamma_\theta + \omega'_\theta)e^{(\gamma_\theta-\omega'_\theta)t}\left[1 - \frac{c_1(\gamma_\theta - \omega'_\theta)(2\beta_\theta + \omega'_\theta)}{2(b^2c_2 + c_1(\beta_\theta\gamma_\theta - ab))}\right] \\ &- 2(\gamma_\theta^2 - (\omega'_\theta)^2)e^{\gamma_\theta t}\left[1 - \frac{\beta_\theta\gamma_\theta c_1}{b^2c_2 + c_1(\beta_\theta\gamma_\theta - ab)}\right] - 2(\omega'_\theta)^2\Bigg) \text{ and}\end{aligned} \tag{4.69}$$

$$\begin{aligned}\Phi_\theta^Y(t) &= C_2\left(\gamma_\theta(\gamma_\theta-\omega'_\theta)e^{(\gamma_\theta+\omega'_\theta)t}\left[1-\frac{c_2(\gamma_\theta+\omega'_\theta)(2\alpha_\theta-\omega'_\theta)}{2(a^2c_1+c_2(\alpha_\theta\gamma_\theta-ab))}\right]\right.\\ &+ \gamma_\theta(\gamma_\theta+\omega'_\theta)e^{(\gamma_\theta-\omega'_\theta)t}\left[1-\frac{c_2(\gamma_\theta-\omega'_\theta)(2\alpha_\theta+\omega'_\theta)}{2(a^2c_1+c_2(\alpha_\theta\gamma_\theta-ab))}\right] \qquad (4.70)\\ &- \left.2(\gamma_\theta^2-(\omega'_\theta)^2)e^{\gamma_\theta t}\left[1-\frac{\alpha_\theta\gamma_\theta c_2}{a^2c_1+c_2(\alpha_\theta\gamma_\theta-ab)}\right]-2(\omega'_\theta)^2\right)\ ,\end{aligned}$$

where

$$C_1=\frac{b^2c_2+c_1(\beta_\theta\gamma_\theta-ab)}{\gamma_\theta(\omega'_\theta)^2(\gamma_\theta+\omega'_\theta)(\gamma_\theta-\omega'_\theta)} \quad \text{and} \quad C_2=\frac{a^2c_1+c_2(\alpha_\theta\gamma_\theta-ab)}{\gamma_\theta(\omega'_\theta)^2(\gamma_\theta+\omega'_\theta)(\gamma_\theta-\omega'_\theta)}\ .$$

We can now use the inverse transform expressions (4.19) and (4.21) to obtain expressions for the actual X and Y reactant perturbation auto-covariance functions for cell lag u in the unified spatial-temporal scenario. These expressions can then be numerically calculated given values for system coefficients.

Consistency proofs

We are able to confirm that results (4.69) and (4.70) are consistent with certain system conditions and assumptions. First we can confirm that they are consistent with initial conditions through two stages of differentiation with respect to time. If we take the resulting differentiated expressions, and set $t = 0$, we find that

$$\begin{aligned}\Phi_\theta^X(0)=0 \qquad & \Phi_\theta^Y(0)=0\\ d\Phi_\theta^X(0)/dt=c_1 \qquad & d\Phi_\theta^Y(0)=c_2\\ d^2\Phi_\theta^X(0)/dt^2=2c_1\alpha_\theta \qquad & d^2\Phi_\theta^Y(0)/dt^2=2c_2\beta_\theta\ ,\end{aligned}$$

which show exact agreement with the specified initial conditions in (4.58) and (4.59).

For our second confirmation, if we remove all time-dependent terms from (4.69) and (4.70) we see that the expressions can be rewritten as

$$\Phi_\theta^X=\frac{b^2c_2+(\beta_\theta\gamma_\theta-ab)c_1}{2\gamma_\theta(ab-\alpha_\theta\beta_\theta)} \quad \text{and} \quad \Phi_\theta^Y=\frac{a^2c_1+(\alpha_\theta\gamma_\theta-ab)c_2}{2\gamma_\theta(ab-\alpha_\theta\beta_\theta)}\ .$$

These agree exactly with the stationary case solutions produced in Section 4.4 (see

Equations (4.28) and (4.29)).

4.5.5 Spatial-Temporal Auto-Covariance Values

Taking the exact auto-covariance expressions above, we can evaluate them for any given set of system parameters. In doing this we obtain a qualitative description of the auto-covariance behaviour for different types of spatial reaction system, and also obtain some quantitative measure of the population structure in the full stochastic system. In all of the numerical evaluations in the following section, we deal with our generic two-species system, using Turing's *instability* parameter (I) as the main tool for investigating our parameter space. We thus choose to set the following parameter values (as used for the standard realisations detailed in Chapter 3) detailed earlier in this chapter in Section 4.4.2.

Sub-critical scenarios

Figure 4.4 shows the time evolution of a selection of auto-covariance values for a sub-critical spatial reaction system. It can be seen that the spatial-temporal auto-covariances grow from zero to reach steady-state values within 20 time units. Figure 4.4 highlights the positive correlation between perturbations of both X and Y reactants that are separated by two, one or no cells (i.e. $u \leq 2$), whereas those separated by a lag of five cells have a negative correlation. The final covariance values attained with this system equate exactly with the stationary scenario detailed earlier (see Figure 4.3).

This growth of temporal auto-covariances towards the stationary state values can be seen in more detail in Figure 4.5. This graph shows the growth in magnitude of correlations for all u values in the 50-cell system, as the evolution time progresses. It can be noted that for very small times (e.g. $t = 10$) correlations are negligible for all but small (i.e. $u < 5$) lag values. However, as the system develops strong long-range correlations are created in the expected five-wave pattern.

We can further investigate the effect of variation in system instability for the sub-critical scenario in terms of the magnitude of the correlations produced, and the time taken to reach the stable steady-state solution. Figure 4.6 shows both of these effects. Studying

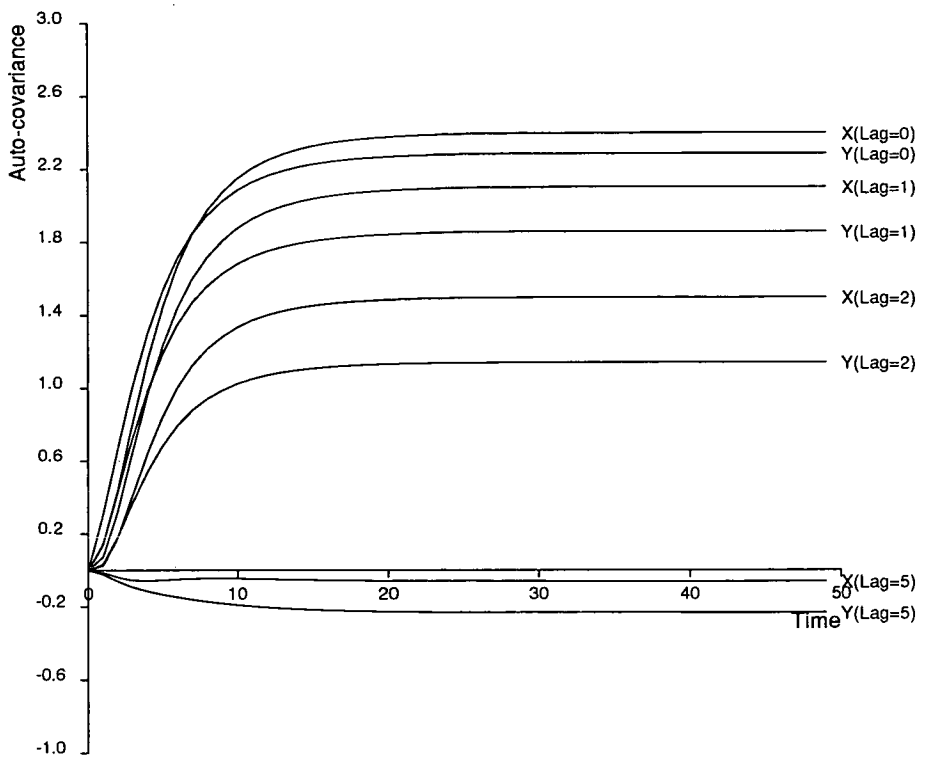

Figure 4.4: Sub-critical temporal auto-covariance values

Using a system instability of $I = -0.1$ this graph shows a selection of temporal auto-correlations for differing cell lag values. The system parameters are set to our standard linearised generic two-reactant spatial reaction system.

the maximum auto-covariance value (i.e. taken for $u = 0$) we see that as the sub-critical instability decreases in magnitude, then the auto-correlation values for perturbations in both species increases, very rapidly so for very small instability values. In addition, we can see from graph (b) in Figure 4.6 that as the stationary state correlation value decreases, so does the time taken to reach stability.

Thus, in the sub-critical scenario, we see that for high magnitude instabilities there are only very short range correlations between perturbations in either reactant type. These correlations are also very small in magnitude, and are established on very short time-scales. However, as the system parameters move towards criticality, we obtain strong correlations between perturbations throughout the spatial reaction system, although such correlations only become fully established over long time-scales (e.g. many hundreds of time units).

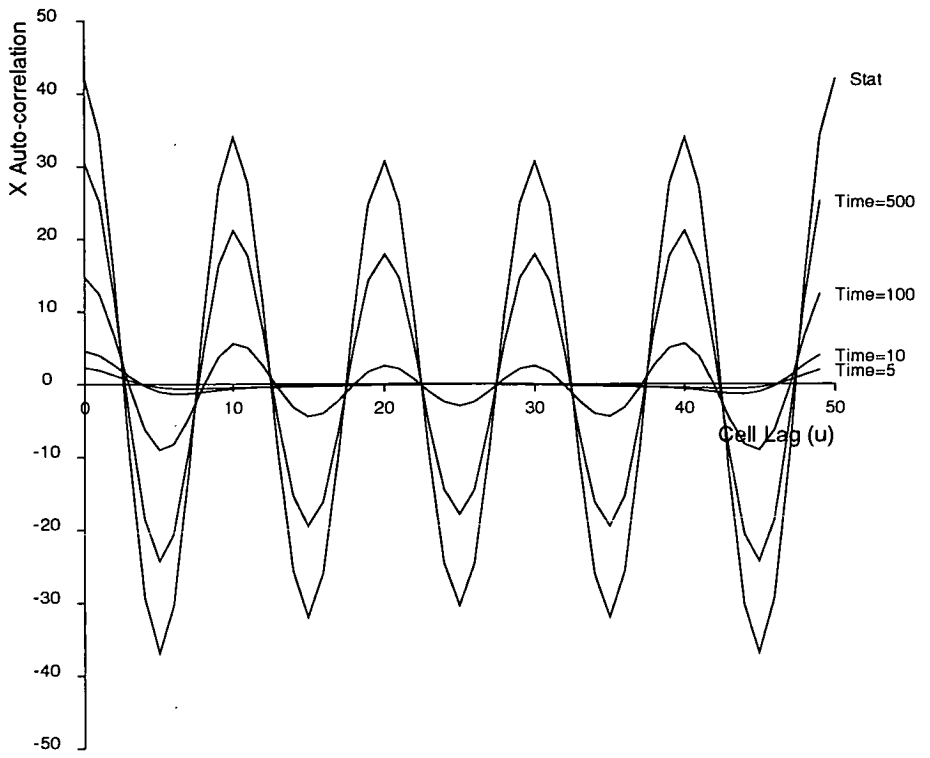

Figure 4.5: Sub-critical spatial auto-covariance values

Using a system instability of $I = -0.001$ this graph shows a selection of spatial auto-correlations for differing time values. The system parameters are set to our standard linearised generic two-reactant spatial reaction system.

Super-critical scenarios

If we now consider the super-critical scenario of our standard two-reactant spatial reaction system, we can again calculate spatial-temporal auto-covariances for both reactants. Here we know that we do not obtain any stationary state for these systems, as all reactant perturbations will tend to move away from equilibrium.

Figure 4.7 shows the spatial auto-covariances for a super-critical system with very small instability ($I = 0.001$). We observe a pattern very similar to that for the sub-critical case in Figure 4.5 — we find long-range correlations that produce a wave-like pattern that grows in magnitude with time. In this scenario, however, the covariances are not limited by the stationary state values, but continue to grow with time.

As we increase the instability for super-critical systems we find that all auto-covariance values become explosively large in finite time-scales. It is only in short times (e.g. $t < 50$) that we can observe interesting effects. However, since it is these time-scales that define eventual system morphology, it is of significant importance to study short-

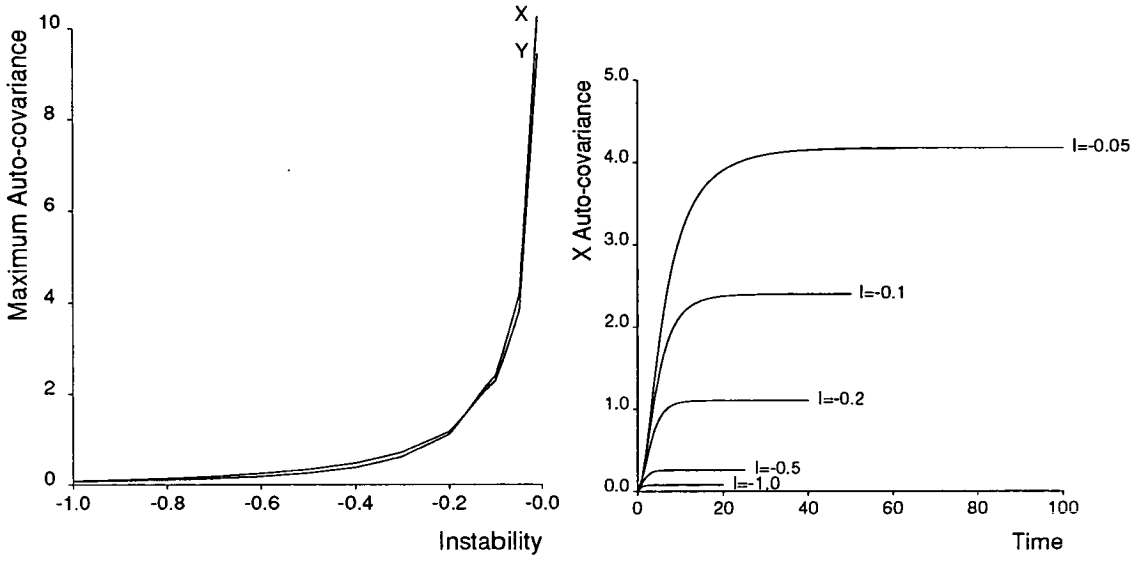

Figure 4.6: Auto-covariance variation with sub-critical instability
These two graphs detail the variation of auto-covariance values with system instability. The left-hand graph (a) shows the maximum auto-covariance value (i.e. the variance, when $u = 0$) with varying sub-critical instability. The right-hand graph (b) shows the rise to stationarity for the X reactant auto-covariance (again with $u = 0$) for a variety of instability values.

term correlation development.

Figure 4.8 shows temporal auto-covariances for a selection of cell lag values for a system with instability $I = 0.1$. It is clear that all covariance values have left the scale of the graph before time $t = 50$, however some very interesting activity can be identified before this time. Firstly, we can observe that there are growing positive auto-covariances for the Y reactant at lags $u = 0$ and $u = 1$; and there are growing negative correlations for X reactant at $u = 10$ and for the Y reactant at $u = 5$. However, more interesting behaviour is found for the X reactant at lags $u = 0$ and $u = 1$. In these cases we see a positive auto-correlation (at small times) reach a maximum value and then turn to a strong negative correlation. A similar activity takes place for the Y reactant at $u = 10$, where an initial negative covariance becomes positive once $t >\sim 28$ time units.

Thus, in this super-critical scenario with high instability, we see that an X perturbation will tend to reinforce itself for up to 46 time units, after which point there will be a strong negative correlation. In contrast, Y perturbations have an ever increasing positive auto-correlation for cells close to the perturbation, and increasing negative correlation for cell perturbations one half wavelength distant.

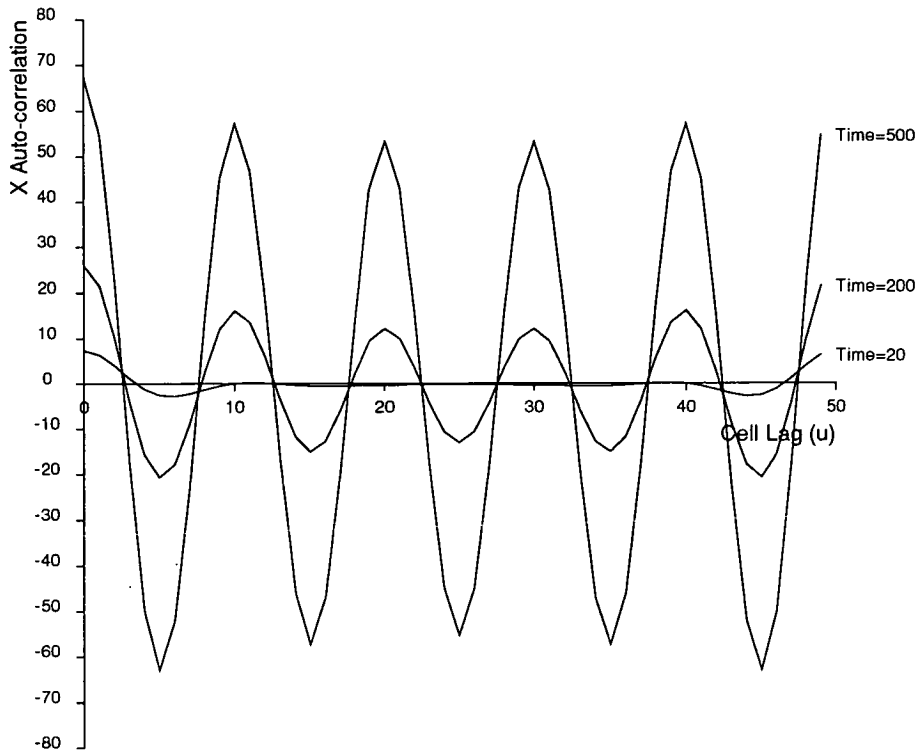

Figure 4.7: Super-critical spatial auto-covariance values

Using a system instability of $I = 0.001$ this graph shows a selection of spatial auto-correlations for differing time values. The system parameters are set to our standard generic linearised two-reactant spatial reaction system. Note the growing strong long-range auto-covariances that follow a five-wave pattern.

We can compare the results in this section to the stochastic simulations detailed in Chapter 5. We see that graphs such as Figure 4.4 and Figure 4.8 and the equations they represent — (4.69) and (4.70) — provide us with a direct measure of the expected development of stochastic perturbations within our spatial reaction systems. The expected lifetimes of perturbations, and the size and range of their expected influence can be used directly by our parallel computer implementations (see Chapters 6 and 7) to ensure efficient and effective simulation.

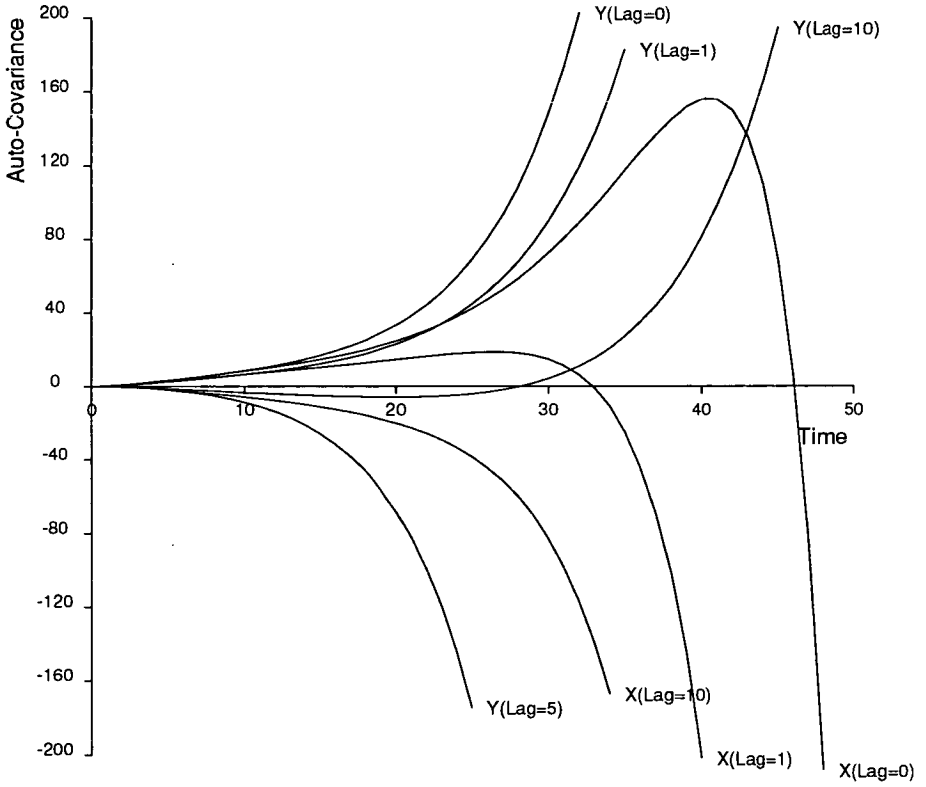

Figure 4.8: Super-critical temporal auto-covariance values

Using a system instability of $I = 0.1$ this graph shows a selection of temporal auto-correlations for differing cell lag values. The system parameters are set to our standard generic two-reactant spatial reaction system with linearised interaction coefficients.

5

Numerical Simulation of Stochastic Spatial Reaction Systems

It is not knowledge but the act of learning, not possession but the act of getting there, that grants the greatest enjoyment.

– Carl Friedrich Gauss

5.1 An Introduction to Stochastic Simulation

As discussed in the previous chapter, increasing attention is being paid to the importance of stochastic systems. It is generally accepted that many physical, chemical and biological systems can be understood fully only by considering their stochastic representations. However, in many fields of science stochastic analysis is still less advanced than its deterministic equivalent, and for this reason stochastic study has often been restricted to computer simulation work (see Morgan [1984]). In the analytic work detailed in Chapter 4 we have provided a first level of understanding for the spatial-temporal development of reaction systems. We found that inherently stable systems, lying near equilibrium, but subject to Gaussian noise perturbations, will produce changes in reactant levels that follow distinct and regular patterns, dependent on basic system parameters. This analytic result shows predicted stochastic behaviour similar to the recent experimental work of Vibert [1994] who observes that random noise activity produces regular frequency behaviour patterns within *neural network* models. In this chapter we detail the results of our stochastic simulations of spatial reaction systems, and thus present the empirical confirmation of our analytic work.

In Section 3.1 we detailed the functionality provided by a suite of computer programs developed to perform deterministic realisations of spatial reaction systems on a variety of computer systems. This software also provides a facility to perform stochastic simulations of spatial reaction systems. In fact, for the majority of the simulations detailed in this chapter, the software is a single program, and by use of command line arguments the realisation can be chosen to be deterministic or stochastic. In addition, we have provided the user with the opportunity to switch between these two modes at run time. This allows investigation of the effects of stochastic development on a persistent system state, as well as the study of the recovery of a deterministic steady state from a previously stochastic scenario.

The development of this software package enables a wide range of simulations to be investigated with relative ease, and the important results of this work are detailed later in this chapter. These results are often substantially different from those obtained in the deterministic regime, and much of our effort will go into highlighting and analysing such departures from the deterministic solution. Later (in Chapters 6 and 7) we will make direct use of both our theoretical and empirical results in order to produce efficient

and effective implementations of stochastic spatial reaction systems. Such studies will highlight, from the programming perspective, an important division between deterministic and stochastic computer realisations of numerical systems — the time taken to compute the solution. The problem of long simulation times is enhanced as we look to study larger, more complex, systems over long timescales. Unfortunately, this is exactly the direction in which we must move if we are to produce results that resemble real-world systems.

When solving deterministic systems, efforts are obviously made to ensure that the numerical solution technique in use is accurate. This often involves using time increments that are small enough to guarantee that the approximations made in the numerical solution technique remain valid. This point aside, the solution technique is generally straightforward, and solution times are consistent and, for most system sizes, small, even when executed on a standard desk-top workstation. However, for stochastic simulations we must employ different solution techniques, and these often prove to be both expensive and unpredictable in terms of computer time. To overcome the former of these problems we can turn to supercomputing resources for our long-term or large-time simulations. However, even there our work can be severely disrupted if we cannot predict the run-time of our simulations. In addition, most available supercomputers are now parallel architecture machines (see Section 5.2) and it can be difficult to achieve optimum performance on such machines, especially for stochastic simulation calculations. We will return to this subject in Chapters 6 and 7, and introduce some of the new implementation techniques and programming algorithms we have developed.

5.1.1 Computer Implementation of Stochastic Simulations

There are now a small number of standard methods for performing stochastic simulations on computers, and these techniques are widespread throughout all scientific disciplines. The basis for all of these techniques lies in the concept of a program generating a series of random numbers to compare against the probability of an *event* occurring within the simulated system. This technique has its foundation in the earliest days of computing in the work of Metropolis *et al* [1953] when it was introduced as the *Monte Carlo* technique — due to the similarity with the generation of a series of "random" numbers on a roulette

wheel. Using this technique we can construct a program such that for a given small increment in time (δt) we have a set of probabilities that certain possible events will occur. We then decide which (if any) event actually does occur by comparison with the generated random number. For example, if we consider a process where a single population of size X can undergo either *birth* or *death* with given deterministic rates λ and μ. We can state that the probability of a birth occurring in time δt is $X\lambda\delta t$ and the probability of a death is $X\mu\delta t$. Thus for each individual member the probabilities of giving birth ($\Pr(B)$) and dying ($\Pr(D)$) are $\mu\delta t$ and $\gamma\delta t$ respectively. Although our spatial reaction system simulations are obviously more complex than this (for example, see Equations (4.3) and (4.4)), the same simulation principles can be applied. Thus for a given δt a uniform random number (u) is generated (in the range 0 to 1) for each reactant. We then make an arbitrary choice that if $u \leq \Pr(B)$ then a *birth* event will occur for that particular individual, whereas if $u > 1 - \Pr(D)$ then the individual will *die*. There are clearly different methods for performing stochastic simulations. For example, we could calculate the expected time until an event occurs, and at that point choose a particular event. However, as will be stressed later, it is vital that our spatial reaction systems simulations retain discrete individuals within the model, each of which can be subject to different interaction probabilities and thus tracked uniquely. In order to allow efficient parallel supercomputer implementations this individual basis for the stochastic models suggests the use of our selected simulation technique.

Our large-scale Monte Carlo processes are computationally intensive for two reasons: they require the generation of very many random numbers; and they need separate calculations to be made for each individual reactant. Execution time *profiling* of almost any stochastic simulation code will show that a large proportion of run time is spent in the random number generation routine. This can lead to a temptation for very simple random number generators which often produce cycles of numbers with very small periods. However, if we are to have any confidence in our stochastic simulation results we must ensure that the random numbers we use to make probabilistic decisions are totally uncorrelated. Much work has been performed in this field, and efficient generators with massive periods are now available from standard texts (e.g. Press *et al* [1988]), and such functions are used in all our simulation work; their high execution-time costs have to be accepted as inevitable. In contrast, the additional cost of performing calculations for individual reactants rather than for whole populations (as in the deterministic scenario)

may seem like something to be avoided. Although it is true that simulation techniques are available that do not require this complete discretisation of system reactants, such methods cannot be used if one considers this work in terms of our final aims (see, for example, the case study in Chapter 8). Since we are dealing with reactants that have certain attributes specific to the individual, and more importantly, the situation when such attributes are evolving in time, we find that we are forced into making probabilistic decisions at the level of the individual reactant.

In addition to the cost of many individual-specific calculations, stochastic simulations are also hindered by the need to ensure that the time increments used are small enough to ensure that only one event can occur in a given time δt. If this were not the case, we could encounter the unrealistic situations of a reactant both *dying* and *reproducing* at the same instant, or alternatively, of a particular event having a probability greater than unity. In the non-linear systems we simulate in this work, it can be seen from Chapter 4 that the probability of events is often related to the current population of a cell. Since these populations are unpredictable *a priori*, and can often become very large, this requires us to use values of δt that are very small, thus ensuring that individual event probabilities always remain small. It has often proved necessary in this work to use values of δt one or two orders of magnitude smaller than those used in the deterministic realisation work — typically $\delta t \simeq 0.0005$ time units. This said, it is nevertheless a straightforward process to implement adaptive simulation time-steps. Thus δt can shrink to accommodate large populations, or can expand when all populations are small.

The algorithm used for the simulation of the development of one-reactant type in a general spatial reaction system is given in Figure 5.1. The terms *birth* and *death* are used purely to assist in understanding the process. It must be noted that for a chemical or biological system such abstract concepts should merely be regarded as reflecting unit increases or decreases in reactant concentration.

5.2 Using Parallel Computers for Stochastic Simulation

Throughout the history of scientific and numerical computing there has been a desire to achieve the *ultimate* performance from a computer. The term "supercomputer" was coined in the early 1970s to describe a machine that performed calculations at a rate way

```
For each cell
    calculate birth and death probabilities
    check probabilities sum to significantly less than unity
    for each reactant in cell
        generate a uniform random number u(0,1)
        compare against birth and death probabilities
        perform necessary births and deaths
    next reactant
    calculate migration probabilities
    for each reactant in cell
        generate a uniform random number u(0,1)
        compare to migration probabilities
        perform necessary migrations
    next reactant
next cell
```

Figure 5.1: Stochastic simulation algorithm

This pseudo-code details the structure of the particular algorithm used to simulate the development of a general spatial reaction system. The terms *birth* and *death* are used in a general sense, and simply represent positive or negative unit changes in reactant levels.

in excess of that achieved by standard computers. The term is probably over-used today as all manufacturers clamber to identify their particular product as a *super*computer. However, one thing that is now certain is that all computer manufacturers are looking towards some form of *parallel* computing architecture for their high performance machines.

Parallel computing is simply the use of multiple processing units for the collaborative solution of a particular problem. For the past forty years computer performance has managed to maintain an approximate ten-fold increase in performance every five years. Although micro-processor performance is still improving, the rate of annual increase has shown a distinct decrease in recent years. This is in most part due to hardware engineers beginning to push against the fundamental laws of physics. Namely, electrons cannot travel faster than the speed of light; they must travel at least some finite distance to effect an operation in a silicon chip, and this distance must be sufficient to allow quantum physical effects (i.e. uncertainty) to be negligible. Thus single processors may become increasingly compact, and hardware research continues to develop mechanisms to enhance processor speed (e.g. reduced instruction set (RISC) architectures, see Patterson & Séquin [1982]). However, we know that there are fundamental limits to the ultimate

performance obtainable from an electronic micro-processor. Thus the only mechanism whereby we can obtain yet more computational performance is to make use of a number of processors concurrently — called parallel processing.

The idea of parallel processing is not new. Most human and natural systems operate in an inherently parallel manner, and it is perhaps unfortunate that early computing machines introduced the notion of sequentialisation for the sake of simplicity. Computer programmers have been stuck with that framework ever since. The quote below from Lewis Richardson [1922] (highlighted recently by Wallace [1988]) dates from over 70 years ago, and shows that parallel thinking seemed to come naturally to a scientist, long before mechanical computers came into existence. The *computers* referred to by Richardson are in fact individuals put to work on a particular arithmetic calculation (with pencil and paper) as part of solving a global weather prediction problem.

> *If the time-step were 3 hours, then 32 individuals could just compute two points so as to keep pace with the weather, if we allow nothing for the very great gain in speed which is invariably noticed when a complicated operation is divided up into simpler parts, upon which individuals specialise. If the co-ordinate chequer were 200km square in plan, there would be 3200 columns on the complete map of the globe. In the tropics the weather is often foreknown, so that we may say 2000 active columns. So that 32×2000 = 64000 computers would be needed to race the weather for the whole globe. That is a staggering figure. Perhaps in some years' time it may be possible to report a simplification of the process. But in any case, the organisation indicated is a central forecast-factory for the whole globe, or for portions extending to boundaries where the weather is steady, with individual computers specialising on the separate equations. Let us hope for their sakes that they are moved on from time to time to new operations. After so much hard reasoning, may one play with a fantasy?*

In order to classify computer architectures a variety of different taxonomies have been developed. Perhaps the most commonly used in the parallel computing field is that from Michael Flynn [1972]. Flynn's taxonomy classifies the available computers as follows, based on the number of *streams* of instructions and of data.

SISD: Single Instruction stream, Single Data stream. This is the theoretical basis

for the simple single processor machine (from Personal Computers to standard mainframes), and is the conventional von Neumann architecture [1963].

SIMD: Single Instruction stream, Multiple Data stream. Here many processors simultaneously execute the same instructions, but on different data. This is the basis for massively parallel machines like the AMT Distributed Array Processor (DAP), or Thinking Machines' Connection Machine. Both of these machines use many thousands of very simple processors and can thereby achieve "supercomputer" performance on certain problems.

MISD: Multiple Instruction stream, Single Data stream. Such a machine would apply many instructions to each datum fetched from memory. No computers that conform strictly to this model have yet been constructed, although there is some debate as to whether the Dataflow machine could fit into this category of Flynn's taxonomy.

MIMD: Multiple Instruction stream, Multiple Data stream. This is an evolutionary step forward from SISD technology. A MIMD computer contains several independent (and usually equi-powerful) processors, each of which executes its individual program. There are several ways of building such a machine — the differences lie in how processors are linked together for communications, and how each is linked to memory.

Of Flynn's classifications, only SIMD and MIMD are currently relevant to parallel computing. In this work we have exploited both of these major architectural variants, and have developed specific parallel implementation techniques for each. These techniques will be detailed in later chapters; in this section we simply introduce the concepts behind the different architectural classes.

5.2.1 SIMD: Data Parallel Machines

The basic architecture of an SIMD computer can be seen in Figure 5.2. A large number of simple processors all execute the same piece of code (broadcast by a host or *front-end* machine) in complete synchronisation, but operate upon the data in their individual memory stores. This approach is *data parallel* computing, using a single instruction

thread for straightforward programming, but using parallelism within the program data to extract high performance. There are simple extensions to standard languages that are usually used when programming such machines. These extensions allow the transfer of data between neighbouring processors, as well as the masking of operations to allow selective execution of some tasks. The data parallel programming paradigm is the basis for much of the current effort in establishing the new High Performance Fortran (HPF) standard (see Koelbel [1994]).

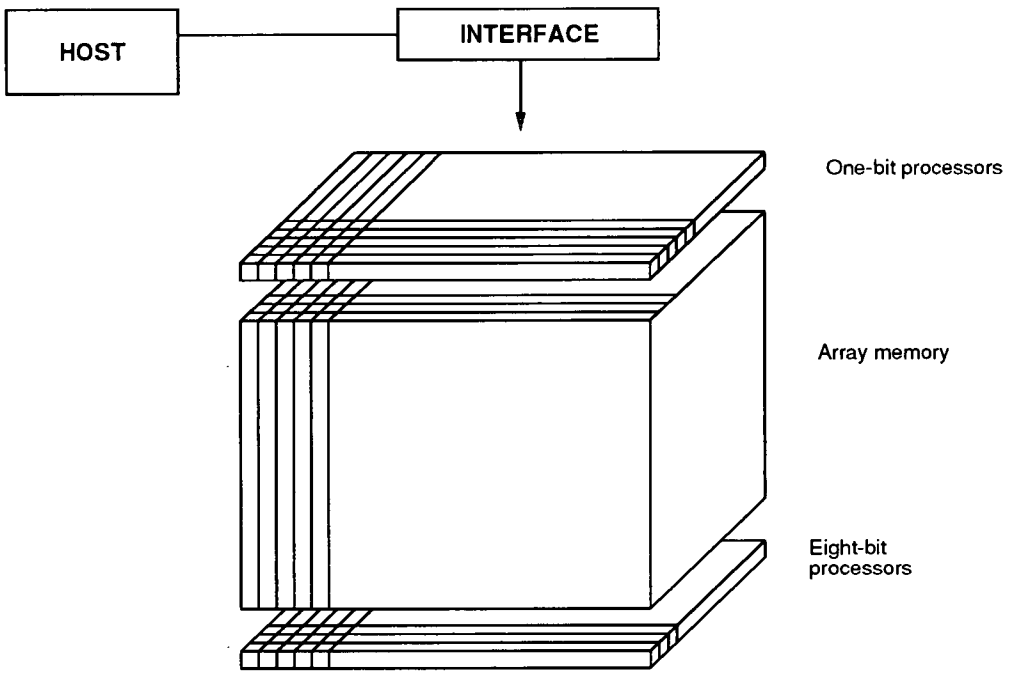

Figure 5.2: Typical SIMD architecture

This diagram shows a schematic layout for a typical SIMD computer. This image is based on the AMT DAP machine, with an array of simple single-bit processors. A separate array of eight-bit co-processors, and columns of *array memory*, are directly available to each processor/co-processor pair.

An example of an SIMD computer is the Distributed Array Processor (DAP) from AMT. The machine consists of a two-dimensional grid of bit-serial processors. In the machine used for some of our early experimental work, there are 4,096 processors arranged in a 64×64 grid with periodic boundary conditions. The machine is hosted from a Sun workstation, and provides a clean development environment with a familiar and stable operating system. The DAP runs a derivative of the Fortran language called Fortran-Plus, which includes all the necessary functions to operate selectively, as well as to transfer data, between processors. A second SIMD machine (the Connection Machine CM-200) has been used for the majority of the two-dimensional data parallel

simulations described later in this chapter and also in Chapter 7. This machine contains 16,384 processors and is also hosted by a Sun workstation; it runs data parallel variants of both C and Fortran.

5.2.2 Shared Memory MIMD Computers

This class of MIMD machine is typified by computers such as symmetric multi-processor high performance workstations, or multi-processor mainframe machines. These systems use standard microprocessors in modest numbers, typically attached to a single memory store by bus-based links. Shared memory machines allow much existing system software to be re-used, as well as implementing many well-understood ideas about managing simple concurrency, e.g. semaphores to manage multi-tasking (see Dijkstra [1965]).

The major draw-back with this class of MIMD machine is that there is an upper limit on the number of processors that can be used with shared processor-to-memory links. A typical shared memory architecture is shown schematically in Figure 5.3. Each processor has a direct link via some bus mechanism to a single global memory store (although increasingly some use is being made of local memory caches). Processors can "communicate" through objects placed in global memory. This is conceptually easy to implement, but has severe implications for the amount of message traffic through the single bus link to memory. There are also problems with memory access control. For example, how to select which process should update an item of memory when two wish to do so concurrently; or the fact that memory locations may need to be "locked" to outside processes when input or output (I/O) is being performed.

Shared memory computers are attractive primarily because they are relatively simple to program. Most techniques developed for multi-tasking computers, such as semaphores, can be used directly on shared memory machines. However, these machines have one great flaw in that they cannot be scaled-up infinitely. As the number of processors trying to access memory increases, so do the chances that processors will be contending for such access. Very quickly access to memory becomes a bottleneck to improving the speed of the computer. Some machine architectures attempt to avoid this problem by dividing memory into as many sections as there are processors, and connecting all segments to processors through a high-performance switching network. However, this

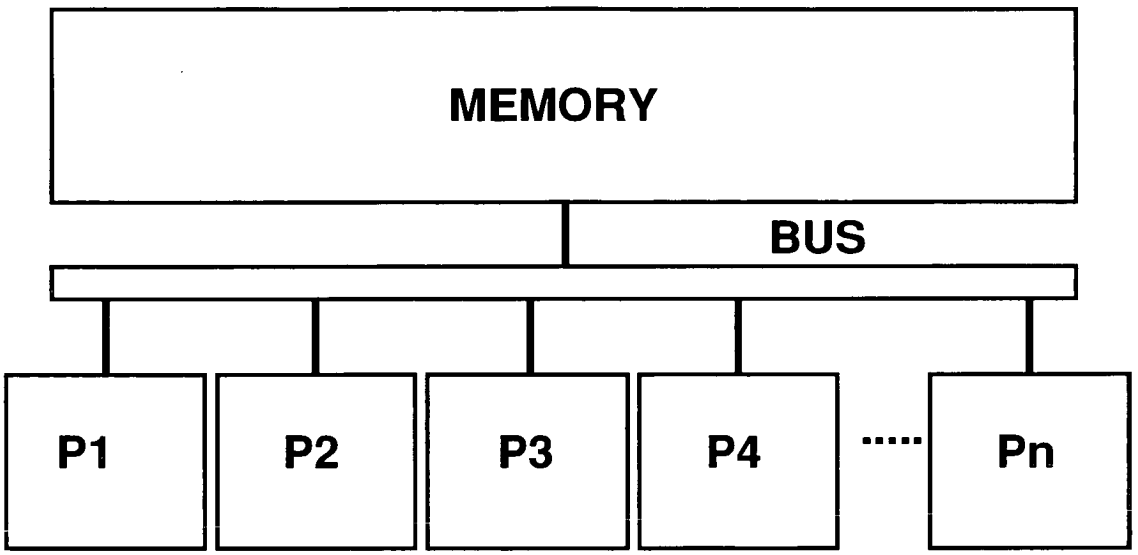

Figure 5.3: Typical shared memory MIMD architecture
This diagram shows a schematic lay-out for a simple shared memory multi-computer. An array of processors is attached to a central memory store via some shared (typically bus-based) link. System software is usually implemented in order to control concurrent read and write access to shared data items.

eventually leads to the same effect and cripples the performance of the switch. The escape route for shared-memory architectures is to introduce memory caches on each processor. This is, of course, simply the first step towards full distributed memory machines.

5.2.3 Distributed Memory MIMD Computers

This class of MIMD computer typically contains machines with much larger numbers of processors, and thus is often thought of as *medium-grained* parallelism, as opposed to *fine-grain* SIMD machines and *coarse-grained* shared memory MIMD computers. Machines in this class can avoid the memory access bottle-necks of shared memory systems by distributing memory, giving some to each processor.

Figure 5.4 shows schematically the architecture of a typical distributed memory machine. The reasons for distributing memory were discussed earlier as the case against shared memory. However we must now consider how to solve the main problem that distributed memory raises — how will the processors be connected and communicate in order to solve their shared task. Connecting all the processors to a bus, or through a single

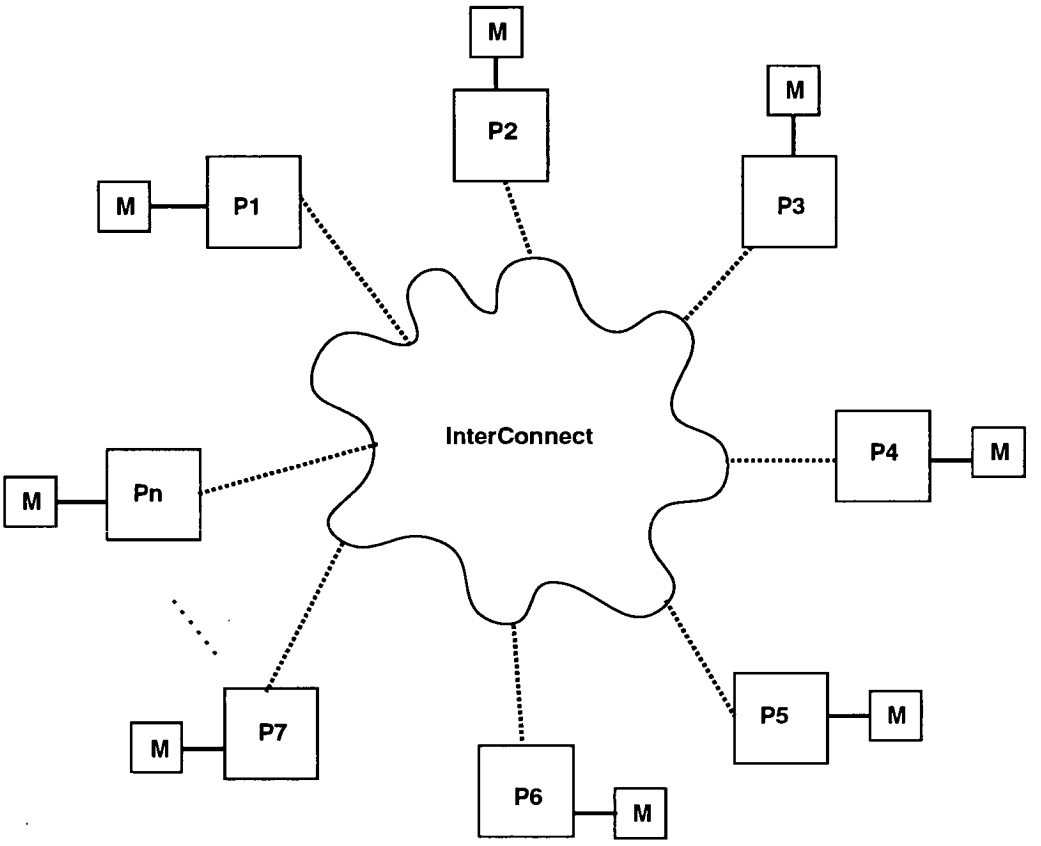

Figure 5.4: Distributed memory MIMD machine architecture
This diagram shows the schematic architecture for a simple distributed memory MIMD computer. Individual processors have dedicated access to a section of the computers memory. Processors are then inter-connected in order to be able to share data values, and to synchronise their operation if required.

switch, brings back the bottlenecks of shared memory systems. Introducing connections from each processor to all other processors is completely infeasible for large numbers of processors, since the number of necessary connections rises as the square of the number of processors, and therefore soon becomes too large to contemplate. The only practical solution comes from connecting each processor to some small subset of the others. Many computers have been built that do exactly this, with a fixed topology of inter-processor links — for example, the hypercube based machines (see Hayes *et al* [1986]). The alternative is to use *switching chips* between processors that allow the user to adapt the topology to suit the particular program being run. This technology was pioneered by Meiko in their Computing Surface machines which are based upon standard four-link Transputers from Inmos (see Smith in Trew and Wilson [1991]). Such four-link devices allow the creation of many topologies such as processor trees, meshes and 4-D hypercubes (see Figure 5.5), which can then be suitably applied to most

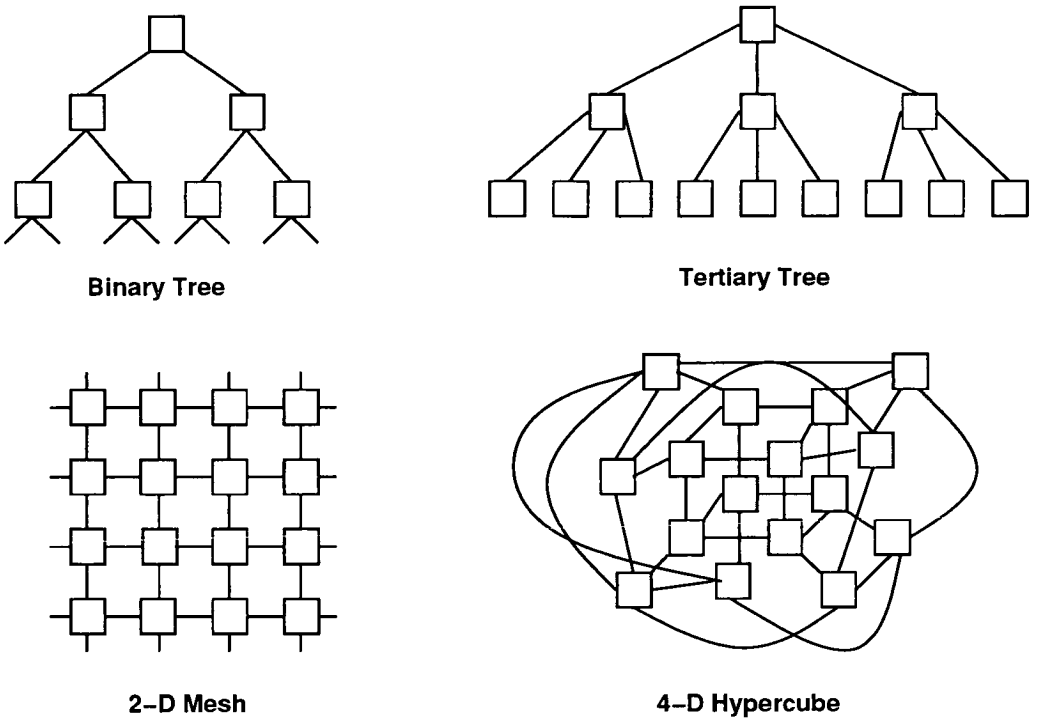

Figure 5.5: Possible topologies with four-link processors

Given independent processors with four available communication links, we can construct a variety of useful multiple processor topologies. Binary and tertiary trees can be used efficiently to solve search or divide and conquer problems. Two-dimensional meshes are ideal for a wide range of physical system simulations, and four-dimensional hypercubes are often used in theoretical physics calculations where all four spatial and temporal dimensions must be simulated.

physical applications. Once a given topology is in place, we then require a programming environment and toolkit to allow efficient use of a distributed memory machine. We discuss these problems and their solution later in Section 5.2.4.

If we accept that the future of general purpose high performance architectures lies in distributed memory multi-computers, then the only problem we face is deciding what development and programming environment is made available on these future machines. We have already seen the manufacturers of transputer-based machines moving away from the Occam language towards providing tools to allow multiple sequential processes, written in C or Fortran, to run in parallel. It seems likely that even further in the future, machines will become available that hide all the fine details of the machine architecture from the programmer. The provision of high bandwidth, low latency, communications systems should enable high-level programming models to become more efficient, thus leading users to a virtual shared-memory machine. This would contain the power

benefits of a distributed memory design, with the ease-of-use of a shared-memory system. Other potential innovations would be an efficient, possibly hardware based, process-to-process message passing system common to all architectures. This would have the effect of making parallel versions of software portable between different machines, with possibly very different architectures. This same goal is currently being explored with the provision of a Fortran 90 standard (see Ellis [1990]) that would provide extensions for parallel data structures, and thus allow porting of code between any machine with a suitable Fortran 90 compiler. While such developments are still being studied, those wishing to use parallel machines must be willing to get involved in understanding the nature of the underlying architecture, and write their software to take best advantage of the features of these systems.

5.2.4 Extensions to the Standard Computing Model

In order to use the parallel architectures detailed above, we must add specialised functionality to our model of computing. Such additions can make parallel computing more complex, and do not always guarantee that extra programming effort will result in commensurate performance improvements. There are three basic types of parallel computer programming:

- In *trivial parallelism* we simply wish to execute a program a given number of times, perhaps with a selection of initial conditions or parameters. In such cases we can use multiple processors running the standard sequential software, but each loaded with the instructions for a subset of the desired executions. This type of parallelism often leads to *task farming* of a problem.

- In *functional parallelism* we identify different operations from within a program, and attempt to apply these operations concurrently on parallel processors. This approach often leads to *pipelining* of an implementation.

- In *data parallelism* we typically run identical programs, but each operates on some subset of our problem data. There are a variety of methodologies to decompose our data set, depending upon the nature of the data and the available architecture. However, they will all generally use the data structures from the original problem

to define the decomposition strategy to be used.

It is clear that all these parallel computing methodologies are centred on the division of a computation between multiple processors, either in terms of problem data, or program functionality.

Decomposition

In general a program can be split into two parts, namely sections that are inherently sequential (e.g. accepting input data from a user), and sections that are potentially parallel (e.g. independent operations on the data). In addition it is often possible to identify separate sequential streams that can be executed concurrently. The main objective of parallelisation is to reduce execution time by *decomposing* the problem to allow parallel execution. It is important to consider how we can identify potential parallelism within a sequential program.

The most frequently exploited source of parallelism is within iterative constructs. Almost every scientific or numerical program contains loops of the sort:

```
Do I = 1, 2000
  Call Operate(I)
End Do
```

Given a large number of processors, each could be programmed to perform the `Operate` procedure upon a particular subset of `I` values. The task of decomposition therefore becomes that of deciding how to divide up the set of `I` values. Potential decomposition schemes may also be identified by studying the data structures within sequential code. There is obviously overlap between these two areas, since programs often contain iterative constructs to loop through the elements of data structures, e.g.

```
Do I = 1, 2000
  Answer[I] = Question[I] ** 2
End Do
```

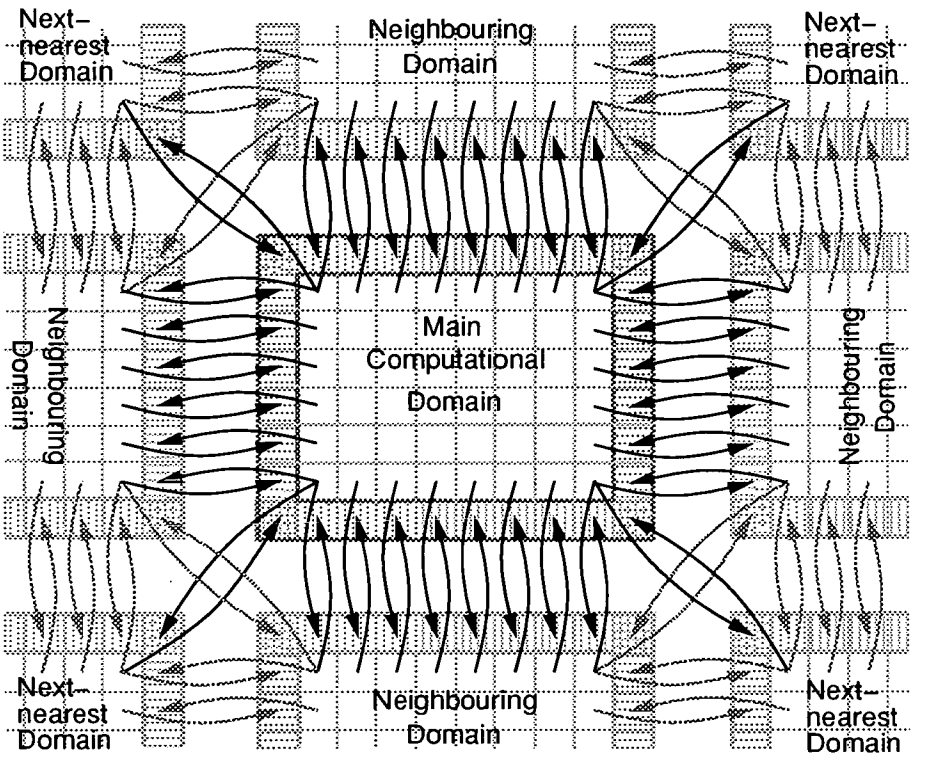

Figure 5.6: Boundary exchanges within a decomposed data structure
A two-dimensional data structure is decomposed to allow parallel solution of an iterative nearest-neighbour process. This diagram shows the communication paths necessary to ensure that the required "boundary" information is exchanged between processes with responsibility for calculations within neighbouring regions.

For these cases of potential parallelism it is important that full consideration be taken of data dependencies within the program. Functions that use the same data will need to communicate new values between each other, and iterative constructs cannot always be simply decomposed. Take, for example, the case

```
Do I = 1, 2000
  Answer[I] = Question[I-1] -2*Question[I] + Question[I+1]
End Do
```

Here all processors will not necessarily have all the information they need to complete the operation, as they will only have direct access to a subset of the current array values. Some communication will therefore be necessary to allow boundary values to be known to all processes. This requirement is shown graphically in Figure 5.6, and illustrates the necessary communications structure for an iterative, nearest and next-nearest neighbour interaction process in two dimensions.

Granularity

On the assumption that a successful decomposition has been performed, we must now consider how efficiently the decomposed program can be executed in parallel. For example, is the best performance achieved through use of maximum parallelism, i.e. using as many processors as possible? Since there is almost always a considerable cost involved with decomposing a problem into a number of sub-problems, this is generally not the case. In addition to the cost of problem decomposition itself (which will usually increase with the number of sub-problems), we must also consider the need to communicate between the separate sub-problems, both to share data as well as to synchronise execution (as in Figure 5.6). Almost all communications have a detrimental effect upon execution time, and thus large message numbers or volumes can severely damage calculation efficiency. It is therefore important to choose the correct *granularity* for the decomposition of the problem. A large granularity (i.e. a small number of large tasks) may not extract all the available parallelism, whereas a small granularity (i.e. a large number of small tasks) may swamp the calculation in communication and decomposition costs.

Load Balancing

The third major consideration for successful parallelisation is whether the sub-problems, allocated to processors as a result of the decomposition, all require the same execution time. If the sub-problems are allocated one per processor, then the total execution time is dependent upon the execution time of the longest individual task — Amdahl's Law [1967]. If this task is inherently sequential (i.e. unparallelisable) it will always provide a measure of the lower limit for execution time, and no additional parallelism can hope to reduce run times any further. Although in such situations the overall performance is limited, efficiency can often be improved by better use of other processors, potentially freeing computing resource to work on other problems (see Figure 5.7). In addition, by placing more than one task on each processor we can potentially reduce inter-processor communication costs.

The best level of efficiency is achieved by having sub-problems spread as evenly as possible across the available processors — called *load balancing*. Achieving this

balance is clearly dependent upon the decisions made during problem decomposition and the size of granularity selected. If either of these are poor then the load balance can rarely recover the efficiency of the implementation. Figure 5.7 shows how to achieve this most efficient use of the available processors by changing the granularity of decomposition.

5.2.5 Parallelisation of Spatial Reaction Systems

From the previous sections we see that optimal parallelisation results from the choice of a good decomposition scheme, coupled with the right granularity to ensure even balance of processor loads across available processors. For spatial reaction systems we can often use the spatial nature of our simulation data to suggest an optimal decomposition. We can thus assign particular regions of our simulation to particular processors in our parallel machine (e.g. as in Figure 5.6). Since the operations to be performed for each location are generally identical, this approach is suitable for both MIMD and SIMD implementations, as detailed later in Chapters 6 and 7.

In deterministic scenarios, when dealing with total reactant populations, the calculation cost for each location is guaranteed to be constant, and we can thus ensure efficient execution by placing equal numbers of spatial locations on each processor. This approach is straightforward, and can produce immediate speed improvements, for example the Connection Machine CM-200 can produce deterministic results at 850 times the rate of a Sun4 SPARCstation (see Chapter 7). However, parallel implementations of such systems are not always required, as calculation times can be relatively short even for large versions of these systems.

Where computational restrictions do tend to occur is with stochastic simulations. As mentioned earlier, me must be prepared for the increased cost of calculating very many random numbers. This suggests that we look to use high performance computers for all our major stochastic simulations. In the early stages of this work we made extensive use of a transputer-based Meiko Computing Surface, a distributed memory MIMD system. Each of the processors in this machine could perform approximately one million floating point operations per second (1 MFLOPS), a performance equivalent to a standard departmental mainframe of the time (e.g. a VAX 750). We were able to make use

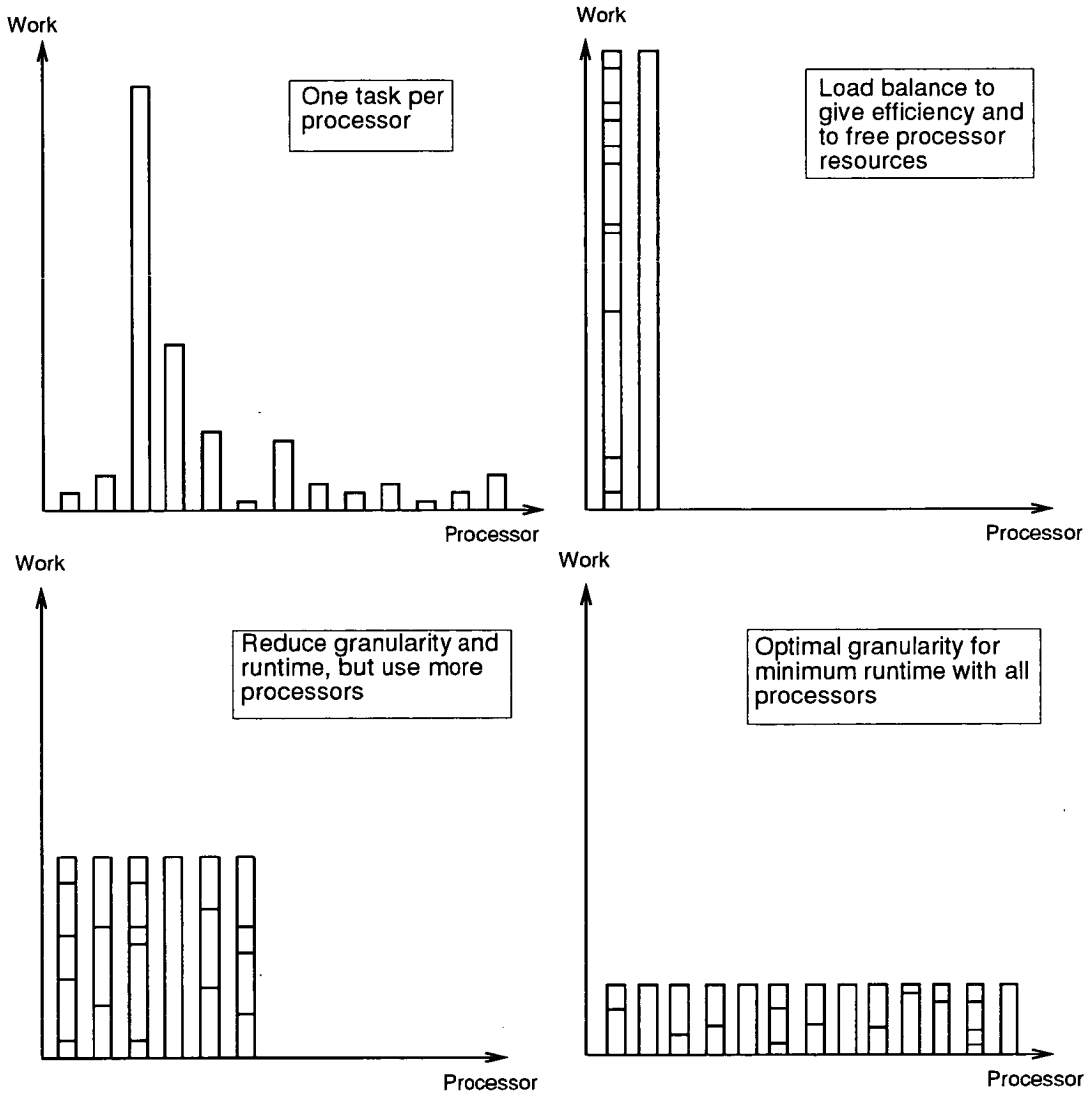

Figure 5.7: Freeing processor resource by using load balancing

A straightforward graphical representation of the inter-play between the mechanisms of decomposition, granularity and load balance. We can improve efficiency with better load balance, and we can reduce overall run-times with finer granularity.

of "domains" with up to 130 of these processors. We therefore had access to a very high performance computing resource; provided it could be efficiently programmed. Transputers have now, of course, been surpassed by RISC-type workstation processors — a single DEC Alpha processor can currently achieve up to 150 MFLOPS performance. However, in our desire to investigate more simulation scenarios, in ever-greater complexity, our parallel programs are written to make use of large numbers of such processors concurrently. In the meantime, the trend for increased computational power continues. For example, the new Cray T3D massively parallel supercomputer at the Edinburgh Parallel Computing Centre contains 256 DEC Alpha processors and can deliver 50 billion floating-point operations per second (50 GFLOPS). We are certain that our future studies of stochastic spatial reaction systems will attempt to make use of the extraordinary power of such systems.

In addition to the load of many random number calculations, since our stochastic simulations are required to track individuals results in a location-based decomposition, we no longer have a guarantee of good load balance for these systems. In fact, from our simulation results shown later in this chapter, we know that often single locations can contain several orders of magnitude more reactants than the average. Such situations can lead to drastic imbalance in processor loads, and hence highly inefficient use of parallel computers. This problem is aggravated in our particular spatial reaction system as it is typically extremely difficult to predict the location, distribution and temporal occurrence and longevity of high population locations (*hot-spots*). It can thus be impossible to design an effective decomposition and load allocation strategy in advance of stochastic simulation. We have therefore found that we must consider *dynamic* methods for load-balancing, whereby computations are allocated (or re-allocated) to processors at run-time, using an adaptive mechanism to ensure optimal load balance.

Dynamic load balancing has been a major issue for parallel and distributed computing, almost since the first such machines were used (see Fox *et al* [1988]). Many of the simulation results presented in this chapter are made achievable, not only by the present availability of parallel supercomputers, but also by the use of dynamic load balancing techniques developed during the course of this work (see Smith and Wilson [1991], Smith [1993], and Smith & Renshaw [1993]).

Chapters 6 and 7 detail two specific load balancing algorithms developed for spatial

reaction systems on MIMD and SIMD architectures respectively, as well as the performance results achieved through use of these techniques. However, in this chapter we now concentrate on simulation results purely in terms of spatial reaction systems, and their relation to our analytic results detailed in previous chapters. Perhaps the most important of these results are those that allow us to extract some link to our analytic understanding of such systems. Given a quantifiable understanding of the nature of stochastic system development, e.g. underlying population growth or decay rates, or the expected distribution of population hot-spots, these results have then been used to confirm parameter inputs to our dynamic load balancing functions. This enables the resulting implementations to automatically perform at near optimal efficiency for any given spatial reaction system.

5.3 One-Dimensional Stochastic Simulations

The majority of the simulation results detailed in this section are based around our generic one-dimensional spatial reaction system. Maintaining consistent values for the majority of system parameters allows greater insight into the fine effects produced by varying the remaining parameters that have been shown to be of earlier interest. Our "standard" system was introduced in Chapter 3, and it is used throughout our investigations, whenever possible. There is obviously an endless amount of simulation work possible, even when restricted to our particular spatial reaction system. Although a great many scenarios have been investigated during the course of these studies, in this chapter we attempt to distill the general features of stochastic simulation results, in order to highlight the important aspects of these systems. Later we will detail results that are more particular to a specific application. Although such specific studies form the basis for the majority of our simulation investigations, full details of all results would prove too expansive to be reported herein.

Our generic scenario contains two reactant types on a ring of 50 sites. Non-linear interaction coefficients are chosen to produce spatial wave structures of finite wavelength (see Section 2.2.3) with a single governing variable I — the system *instability*, as discussed in Section 3.1.2. Variation of I allows us to analyse the stochastic system in the three criticality states: with $I < 0$ we have sub-critical behaviour (in which a

deterministic realisation produces morphologically stable waves with amplitudes that decay exponentially towards equilibrium); with $I > 0$ we have the super-critical scenario of permanent deterministic waves; or finally, at criticality ($I = 0$), we have no governing eigenvalue, and the final system state is dependent upon initial conditions. The only other system parameter to be varied here is the system periodicity (λ') which acts as a specification of the number of reactant waves expected around the ring. In effect this parameter controls the migration rates for each reactant, with increase in the numbers of waves corresponding to a decrease in migration rate. It will be seen later that the discrepancy between deterministic and stochastic patterns can vary substantially with the chosen migration rate.

All of the results presented in this chapter relate to simulations of the full non-linear specification of our spatial reaction system. This choice is made on the grounds of accuracy of simulation; since our Monte Carlo techniques are a numerical calculation of the stochastic solution for the system, there seems little point in using additional approximations via linearisation of the interaction functions. Therefore, although linear realisations were studied as a comparison exercise for the deterministic regime, we feel that a similar comparison for the stochastic simulation case is of less worth. In addition, there are conceptual problems involved with simulating a linearised system in which we study the evolution of small perturbation rather than actual populations. As was seen in the deterministic case, linear realisations produce reactant "populations" that can be negative, since perturbations from equilibrium are allowed to explode in both positive and negative directions. However, when dealing with stochastic simulations, we must be much more aware of the *reality* of the system. Since we deal with unit changes in reactant populations or concentrations, once these reach zero, there can be no possibility of values then going negative.

5.3.1 Short-Term Critical Simulations

The first area of interest for our stochastic simulations, is the behaviour of the reactant populations in the short time period following initialisation of the one-dimensional system. We know from our deterministic results in Chapter 3 that initial conditions can have a direct bearing on our final system state. This is either because the system

is at criticality, or if (in either super- or sub-critical scenarios) any initial perturbation from equilibrium will lead to diffusion driven instability to create spatial waves in the cell populations. In all deterministic scenarios, if there is no initial perturbation from equilibrium, then no changing effects will be observed. In contrast a stochastic system will always be able to move away from equilibrium.

Let us first look at the development of a stochastic critical system within the first time unit of evolution. Figure 5.8 shows four graphs of reactant populations in such a system, and they show a highly random state at early times ($t = 0.2$). However by $t = 0.4$ we start to see the stochastic populations settle down. Thus in the second graph we see the X- and Y-populations beginning to match each other in the individual cells, although any "wave" structure that exists has very short wavelength and is difficult to discern. As we move on to $t = 0.8$ we observe the first population *explosion* (in cell 35), and we can clearly see the beginnings of a stochastic wave pattern. This pattern has become quite obvious by the last graph ($t = 1.0$) where we can identify five major wave crests around the ring, although their locations are somewhat irregularly spaced.

It is interesting to study the location of wave-crests in the development of this critical simulation. At time $t = 0.8$ we observe that five large populations have grown within each ten-cell region roughly bounded (in this particular simulation) by low populations in all cells that are numbered as a multiple of ten. However, from the slightly longer term graphs in Figure 5.9, by $t = 3.0$ we see the morphological structure changing substantially. For example, at $t = 3.0$ cells 0 and 20 now host large population concentrations, and we have a clustering of four peaks between cells 20 and 35. As the simulation progresses, the wave structure appears to move through completely unrelated morphologies — the graph for $t = 4.0$ in Figure 5.9 shows just four main reactant hot-spots now centred on cells 11, 26, 42 and 48.

The above results cover the short- to medium-term development of stochastic waves at criticality. We would expect such systems to contain stochastic wave structures that are somewhat unpredictable, there being no leading eigen-value to *drive* the solution. This belief is born out by the simulation results shown, where we observe systems that undergo large variations in the number of waves present at the start of the simulation.

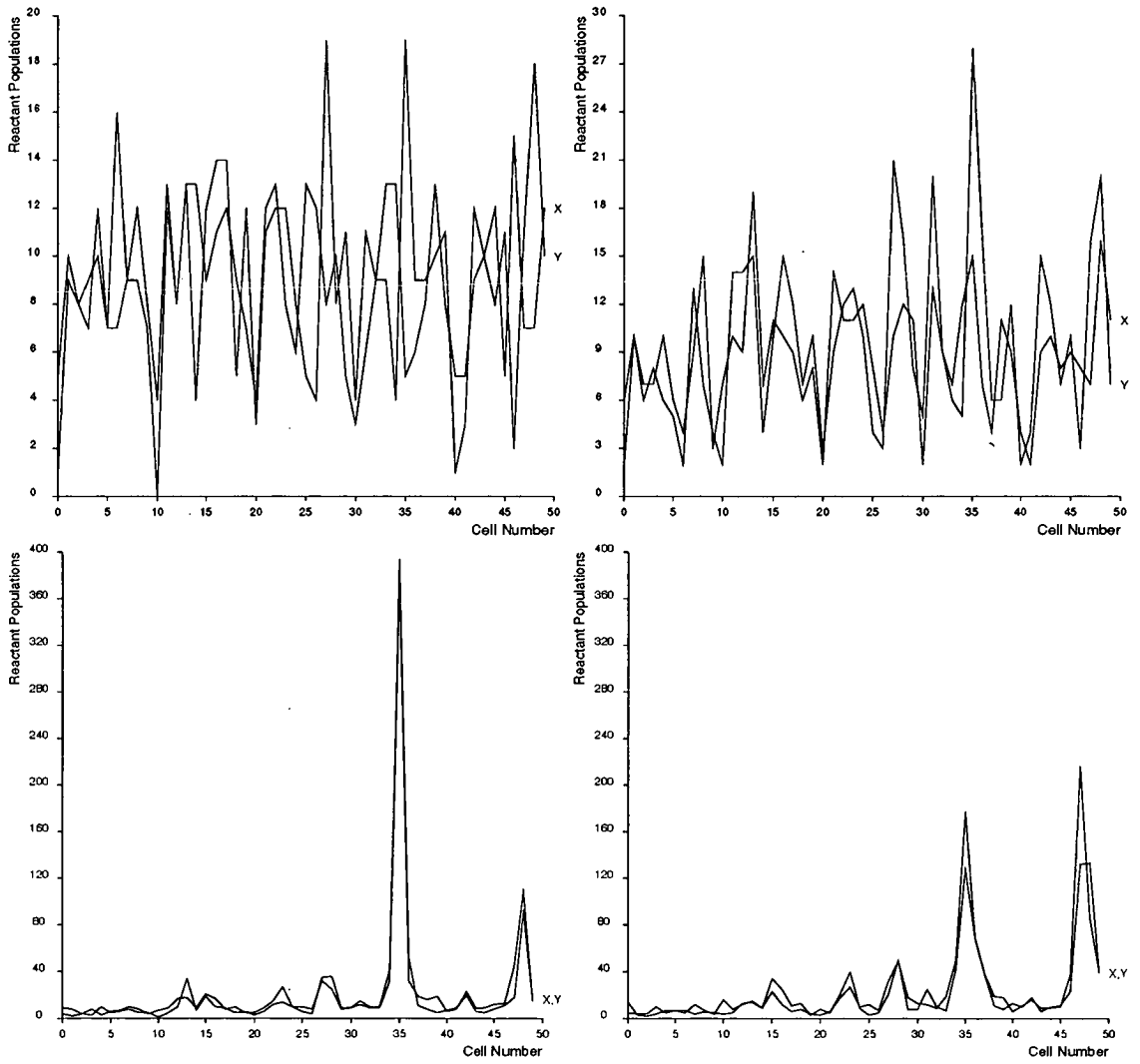

Figure 5.8: Short-term stochastic wave development at criticality

These four graphs show reactant populations at various stages of stochastic system development. The system has non-linear critical interactions ($I = 0$). The graphs show the reactant populations at times $t = 0.2$ (top left), $t = 0.4$ (top right), $t = 0.8$ (bottom left), and finally the emergence of clear wave structures at $t = 1.0$ (bottom right).

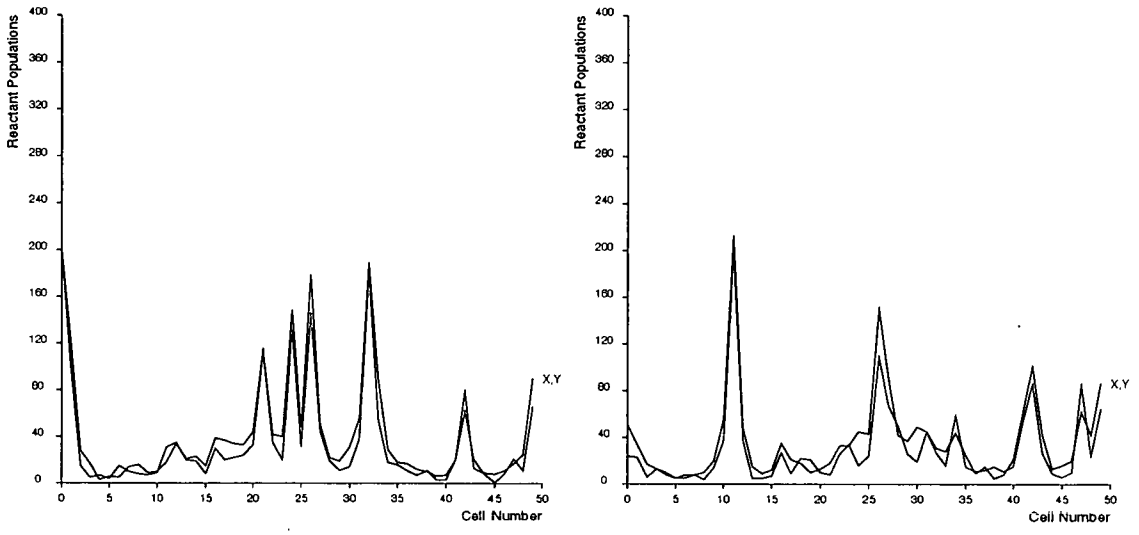

Figure 5.9: Stochastic wave departure from initial perturbations
These two graphs show reactant populations at medium-term stages of stochastic system development. The system has non-linear critical interactions ($I = 0$). The graphs show the reactant populations at times $t = 3.0$ (left) and $t = 4.0$ (right).

5.3.2 Short-Term Non-Critical Wave Structures

Let us now examine the short-term behaviour of our standard system with non-zero instability values. In these simulations we have chosen to use strong initial conditions, with five equally spaced cells having initial populations of $X_{r'} = 15$ and $Y_{r'} = 15$, and all other cells with $X_r = X^* = 10$ and $Y_r = Y^* = 10$. Here $r' = 0, 10, 20, 30, 40$, and from our deterministic results we would expect a wave structure to form with peaks at these locations. Figure 5.10 shows the temporal development of such a system in a super-critical state ($I = 0.1$). This graph is drawn in three dimensions, and shows only the population of reactant X, since the spatial-temporal distribution of both reactants follows the same general pattern (as can be seen in Figures 5.8 and 5.9). The vertical axis markings have been removed from Figure 5.10 to assist with clarity, and the X-population maximum is 1,500. We therefore see that our super-critical system undergoes only minor development during the first time unit, but then explodes into action. The graph details the emergence of five clear stochastic wave crests, and despite the opportunity in the first time unit to lose any effect of the initial conditions, we see that four of these hot-spots lie close to the "expected" location.

Although mesh-based representations of the time development of our simulations are useful for viewing the scale of any wave structures, in order to identify crest location

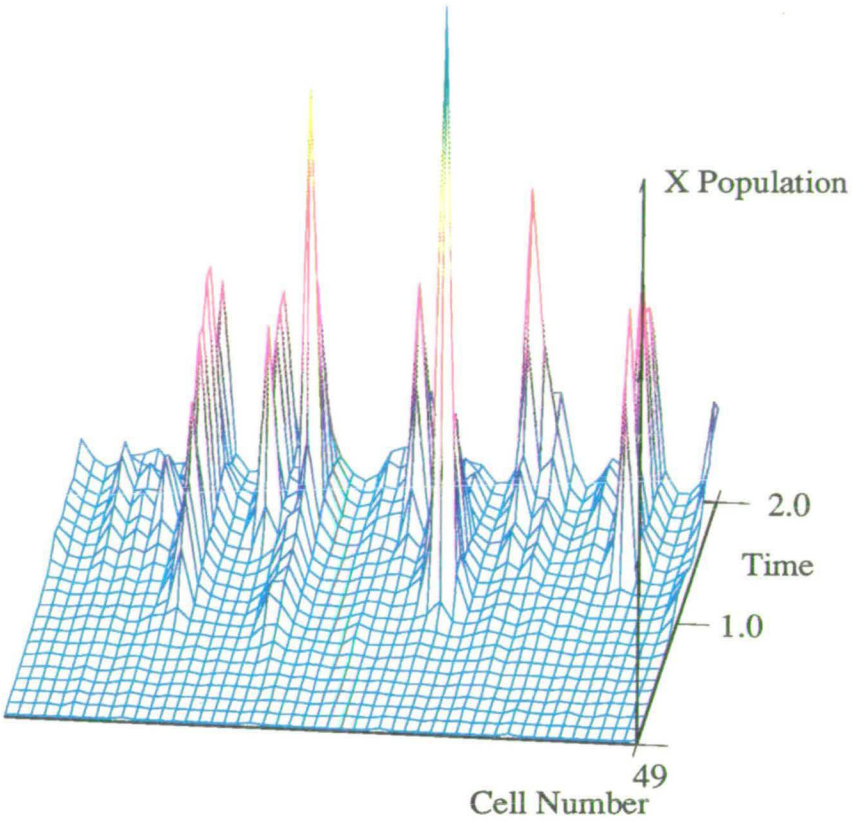

Figure 5.10: Early time development of super-critical stochastic waves
The population of reactant X within a 50-cell ring is shown for a short-term stochastic simulation, following an initial state with five positively perturbed cells. The system has non-linear super-critical interactions ($I = 0.1$). The graph shows the emergence of five clear wave crests, distributed fairly evenly around the ring. It is interesting to note the temporal persistence of these structures. The maximum value on the vertical axis corresponds to an X-reactant population of 1,500.

and temporal persistence more accurately, we feel that a contour plot is more useful. Figure 5.11 thus provides a direct comparison of short-term simulations for the super-critical scenario detailed above, along with similar plots for the sub-critical and critical systems. There are a number of features of these graphs that are worthy of mention:

- The super-critical system develops stochastic waves that are much more regularly located then either the sub-critical or critical cases.

- There is distinctly less stochastic "activity" in the sub-critical scenario. Hot-spots certainly do occur, but the population levels are lower than in either critical or super-critical scenarios. In fact, following the few initial population explosions between $t = 0.5$ and $t = 1.5$, the system seems to have returned to a very subdued

state by $t = 2.0$.

- In the deterministic sub-critical case any initial perturbations always return to the equilibrium state. Although in the stochastic version we have a decay of population hot-spots that is obviously faster than in the super-critical case, we should note that there still exists a clear five-wave structure, albeit with low population levels, and new population centres continue to develop (see Section 5.3.3 below).

- Through comparison with the super- and sub-critical simulations, we now have an enhanced understanding of the important features from the critical scenario. Figure 5.11 shows that at criticality, although we have many examples of persistent stochastic wave crests, their number and location bears far less resemblance to the expected five-wave structure. For example, at $t = 1.8$ we appear to have six or seven stochastic waves in three main clusters around the ring. This gives yet more evidence that the critical case does not follow exactly the same morphological structure "expected" from the system parameters.

5.3.3 Long-Term Stochastic Wave Structures

Using our high performance computing resources we are able to run our standard stochastic simulations over longer time periods. For example, simply the use of 20 SPARC workstation processors (whether connected via a local area communications network, or within the Meiko CS-2 SPARC-based Computing Surface) increases our simulation time substantially, even if poor load balance leads to only 50% efficiency. In addition, efficient use of the SIMD architecture CM-200, or the MIMD CrayT3D, provides computing power well over two orders of magnitude greater than a powerful single workstation. Direct use of these resources this allows us to add to the results detailed in Section 5.3.2, which covered the early formation of wave structures.

In general we see that we generate wave structure patterns that follow logically from these earlier results. It must be stressed that it is difficult to make definitive statements about the results of these stochastic simulations. By their very nature they exhibit strong random fluctuations in their structure, and we should never expect to observe and define regular emerging patterns. What we are therefore looking for is some general

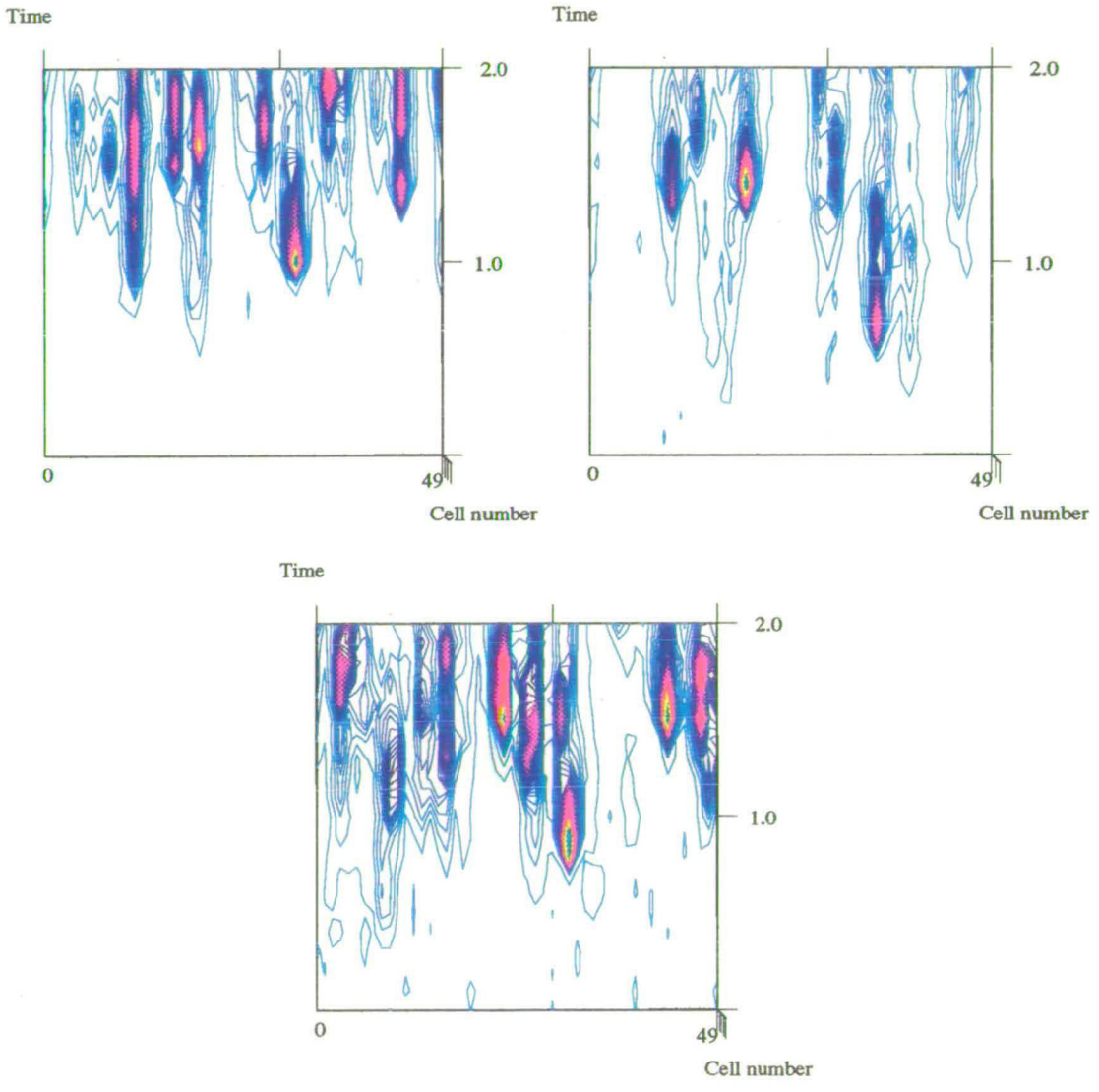

Figure 5.11: Variation of stochastic wave development with criticality
These contour plots show the spatial and temporal development of stochastic wave structures in a one-dimensional spatial reaction system. The system has non-linear interactions, and the three graphs represent the scenarios of super-criticality ($I = 0.1$, top left), sub-criticality ($I = -0.1$, top right), and criticality ($I = 0.0$, bottom). Contour colours use a "rainbow" look-up table and show reactant levels between zero and 2,000. Blue represents populations below 200, red those between approximately 500 and 1,000, yellow represents populations between 1,000 and 1,500, and green colours show those cells with more than 1,500 reactants.

description of the behaviour, and thus some extraction of qualitative features. In the following sections we will therefore describe particular details that we feel give an insight into the general system activity. It is encouraging that in doing this we see that the results can be matched to the stationary, and long term temporal covariance analyses produced in Chapter 4. In particular we observe that for systems with low magnitude instabilities we obtain wave patterns with many regular peaks, whereas for high sub- and super-critical systems hot-spots are more irregularly located as well as being less numerous.

Figure 5.12 shows some long-term snap-shots of a system that is close to criticality, with $I = -0.001$. These graphs show the consistent presence of between five and six roughly equal-strength wave crests. Although some of these hot-spots may change location, they are generally fairly consistent. For example, clear crests can be seen close to cells 4, 25 and 46 at all given times, and close to cells 14 and 35 on three of the four graphs. While these wave patterns certainly retain their stochastic nature (e.g. at time $t = 40$ we could postulate the presence of 10 wave crests) they are distinctly more regular and wave-like than those shown later (Figures 5.13 and 5.14). This is the result we would expect from our (linearised) stochastic analysis of these systems (see, for example, Figure 4.1, where regular global correlations should exist throughout the system).

In comparison, we can look at our spatial reaction system simulations with a high magnitude sub-critical instability $I = -0.1$. Thus Figure 5.13 details the evolution of a rather different system in some important respects. We now see a much wider variation in the number and location of strong wave crests, and the formation of a more inhomogeneous system, with localised areas of the spatial domain becoming either highly active or relatively quiet in terms of reactant hot-spots.

The sub-critical analytic results detailed in Chapter 4 predict that systems such as that covered by Figure 5.13 should posses only short-range correlations between reactant populations. (see Figure 4.3) Thus activity effects close-neighbour locations only, and so no global patterns are to be expected. The graphs in Figure 5.13 exhibit this behaviour almost exactly, although again it is difficult to make definite quantitative claims for these simulation results, and we must be willing to see qualitative results. We see nine irregularly located waves at time $t = 10$, five when $t = 25$, and eight at $t = 50$. In

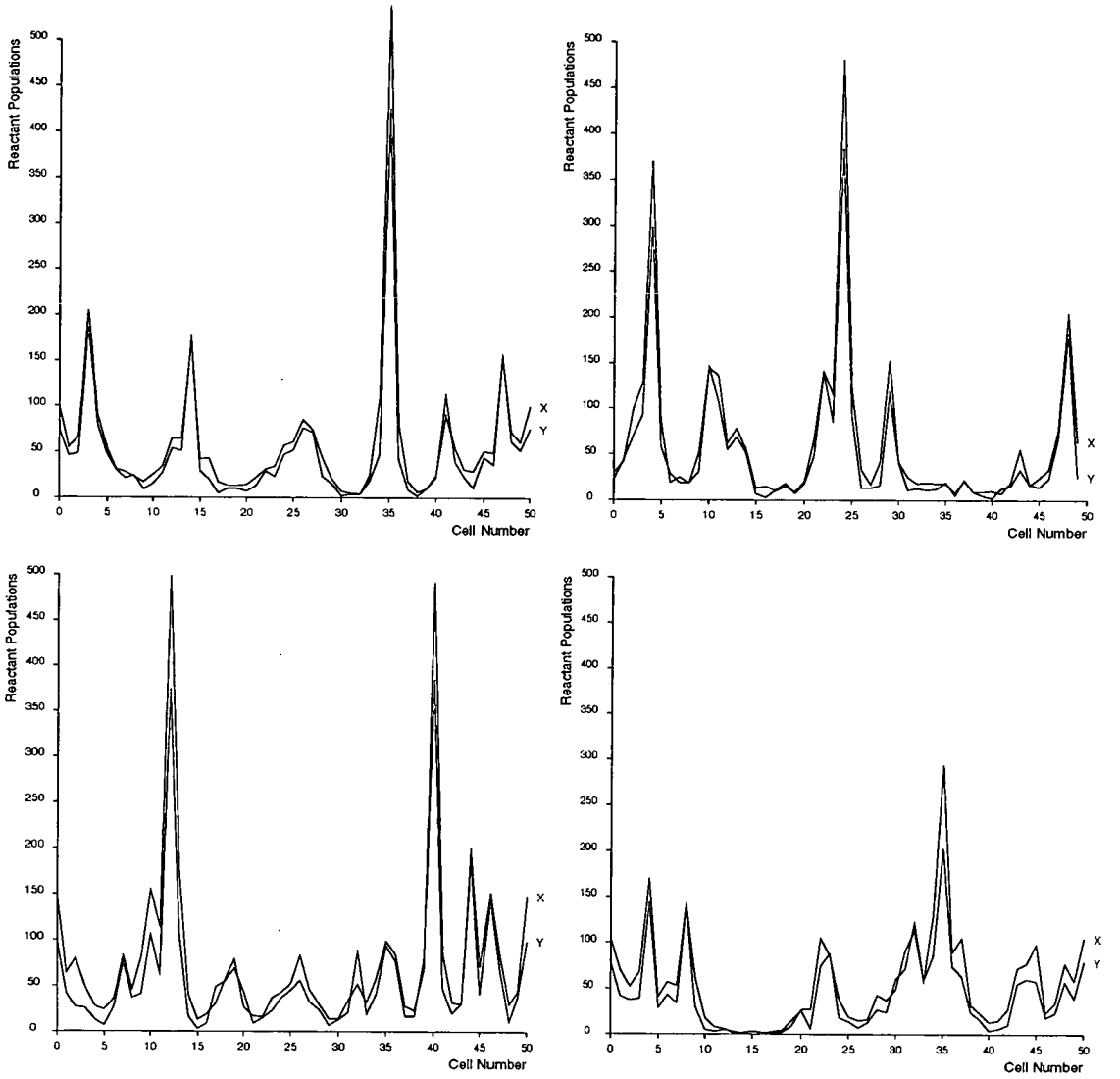

Figure 5.12: Long-term stochastic waves close to criticality

These four graphs show reactant populations at various stages of stochastic system development. The system has non-linear slightly sub-critical interactions ($I = -0.001$). The graphs show the reactant populations at times $t = 10$ (top left), $t = 25$ (top right), $t = 40$ (bottom left), and finally at $t = 50$ (bottom right). Note the clear presence of numerous regular stochastic waves.

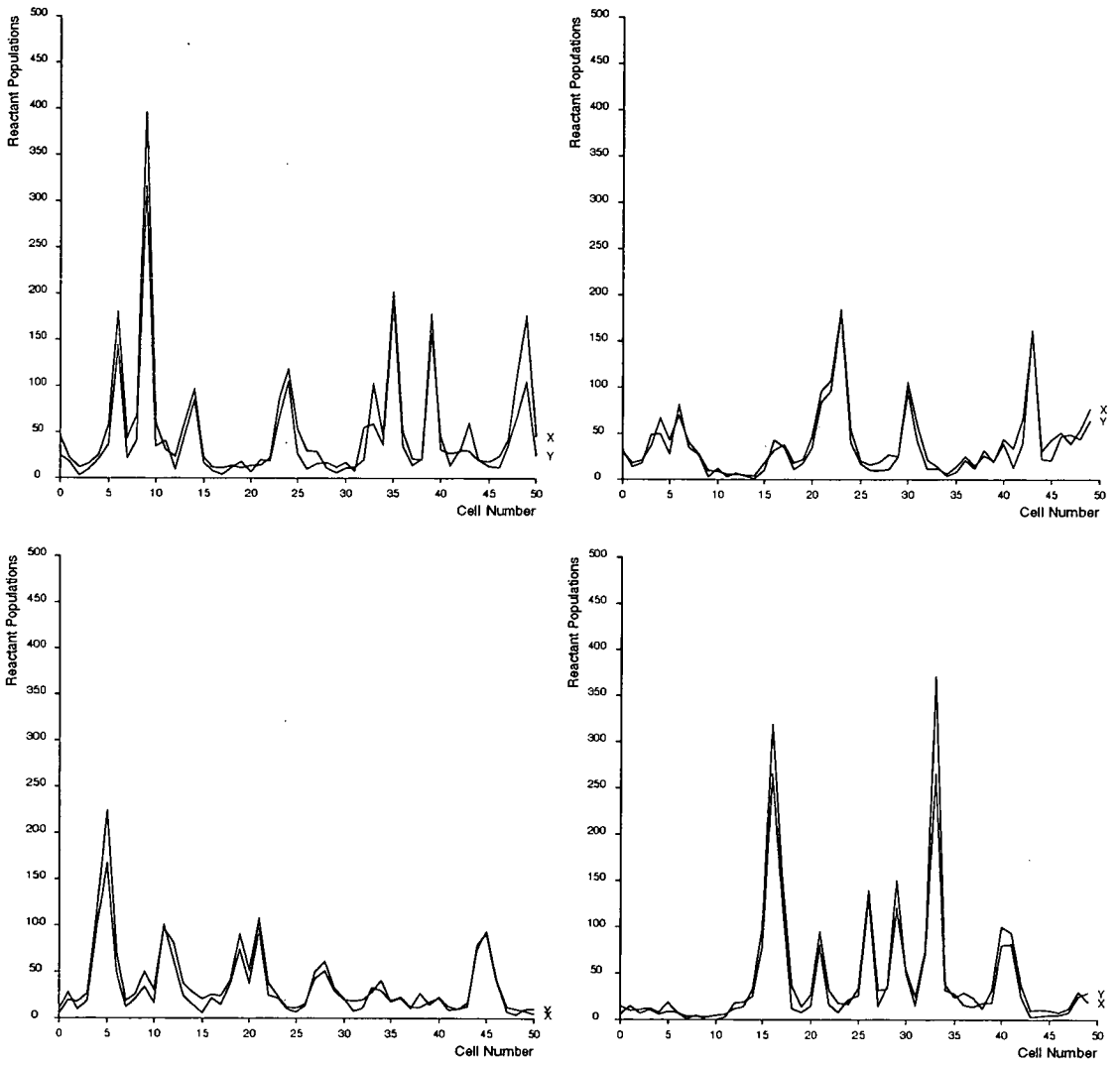

Figure 5.13: Long-term sub-critical stochastic waves

These four graphs show reactant populations at various stages of stochastic system development. The system has non-linear sub-critical interactions ($I = -0.1$). The graphs show the reactant populations at times $t = 10$ (top left), $t = 25$ (top right), $t = 40$ (bottom left), and finally at $t = 50$ (bottom right). Note the presence of varying numbers of stochastic waves in an inhomogeneous distribution.

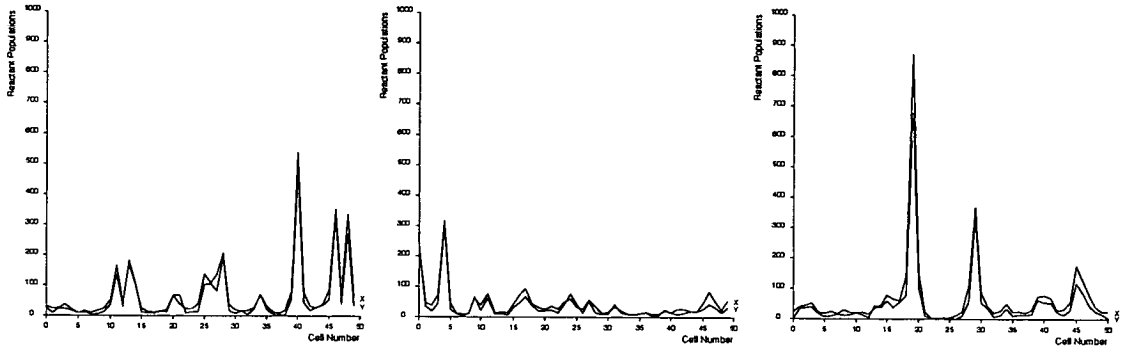

Figure 5.14: Long-term super-critical stochastic waves

These three graphs show reactant populations at various stages of stochastic system development. The system has non-linear super-critical interactions ($I = 0.1$). The graphs show the reactant populations at times $t = 10$ (left), $t = 15$ (centre), and at $t = 20$ (right). Note the presence of varying numbers of stochastic waves, in a very irregular and inhomogeneous distribution.

addition, the wave crest location varies from relatively regular at $t = 25$ and $t = 40$ to the very inhomogeneous structure at $t = 50$ where all high population activity lies between cells 16 and 40, with the rest of the domain almost empty of reactants. We therefore see that this type of system is much more dynamically imbalanced than those close to criticality, and will therefore potentially require more effort to load-balance on a parallel computer. In Chapter 7 we require such a specification of reactant population homogeneity in order to decide on optimal dynamic load balancing strategies.

As a final example of this variation of stochastic wave patterns with criticality, we can examine the super-critical system with relatively high instability $I = 0.1$. Figure 5.14 shows a number of snap-shots of this system's development. As in the previous case, we again observe very irregular wave structures, this time varying from six, to two, then to four major waves between times $t = 10$ and $t = 20$. In the super-critical scenario, cell populations are highly susceptible to very large population explosions (e.g. see cell 19 at time $t = 20$). The graphs in this case are therefore drawn to a different scale than the previous examples, however this does not change their important features, it simply increases the imbalance and irregularity of this scenario.

Rather than the expected five constant waves, Figure 5.14 details a system that moves from six high population locations at $t = 10$, to just two (cells 0 and 4) at $t = 15$, and to a system with one very large population in cell 19 plus a small number of other hot-spots at time $t = 20$. It can be seen that there is very little regularity in the location

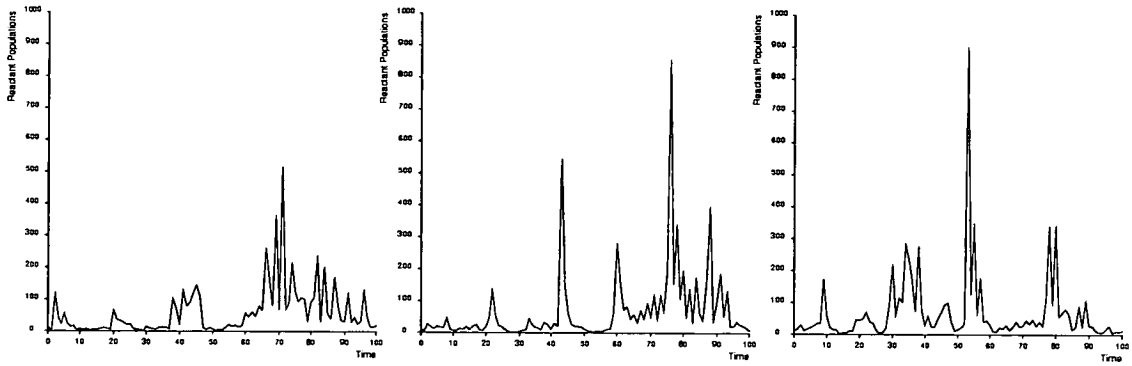

Figure 5.15: Temporal evolution of reactant populations

These three graphs show X-reactant populations for cells 1, 21 and 41 during 100 time units of stochastic system development. The system has non-linear super-critical interactions ($I = 0.1$).

and number of super-critical hot-spots, as predicted by our stochastic analysis (see Figure 4.3 in Chapter 4), and we therefore have a system that will require regular and powerful dynamic load balancing in order to allow effective simulation.

The analytic results shown in Figure 4.8 detail the expected temporal covariance for super-critical simulations with $I = 0.1$. We can show the accuracy of this analysis by comparison the the time evolution of particular cell populations in the above simulations, rather than by studying the particular full system state graphs given above. We therefore see in Figure 5.15 a selection of temporal developments for particular cells (numbers 1, 21 and 41) selected at random from a super-critical simulation.

The important feature to observe in Figure 5.15 is that strong reactant population activity seems to occur in short *bursts* generally separated by longer spells of consistent low populations. The time in between these bursts is obviously highly random, but averages at around 30 time units. The linearised analysis of this system (Figure 4.8) predicts a strong correlation between X-reactant populations in the same cell over a timescale of some 40 time units. We therefore feel that such stochastic analysis can provide a general quantitative specification of hot-spot activity in such scenarios. In the dynamic load balancing algorithm detailed in Chapter 6 we are be able to make direct use of this measure of "time between explosions" in order to optimise the use of these techniques.

5.3.4 Periodicity Variation Simulations

In addition to the study of short- and long-term stochastic waves in different criticality regimes, we can also study the effect of varying the wave periodicity (λ') in short-term simulations. Figure 5.16 shows the contour patterns produced using system periodicities of 0.04, 0.06, 0.1 and 0.2, which create 2, 3, 5 and 10 deterministic waves, respectively.

The main point of interest from these graphs is the vast difference in wave definition as we increase the system periodicity (i.e. decrease the reactant migration rates, see Equation (2.10)). For high migration rate simulations we observe a very complex pattern in the population contours, although the colour scheme does allow us to postulate two or three general regions of high population for the 0.04 and 0.06 periodicity cases. We see a marked contrast between these two cases and those with lower migration rates (the five- and ten-wave cases). Here there is very little stochastic activity shown by the contour maps, other than around definite population hot-spots.

These results suggest that *observable* (i.e. strong) stochastic waves are best obtained by using low migration rates, and therefore large numbers of wave crests. The irregular but slightly more homogeneous patterns produced when the migration rate is high highlights an important departure of the stochastic simulations from the numerical realisations of the corresponding deterministic systems. However, we can also see that the potential imbalance is greatest with low migration rates (high periodicity), and thus in these situations we will need to make most use of efficient dynamic load balancing routines. This is an effect similar to that observed in Section 5.3.3 for high magnitude instability scenarios. In particular, use of system periodicity as a measure for reactant population homogeneity is an important ingredient for the effective use of the dynamic load balancing algorithm introduced in Chapter 7 for SIMD systems.

5.4 Two-Dimensional Simulations

The vast majority of our stochastic simulation studies in two dimensions have been performed as specific ecological model studies, or as investigations into the performance of dynamic load balancing techniques. These results are therefore covered in some detail in later chapters. In particular, the ecological simulations are detailed in Chapter 8 and

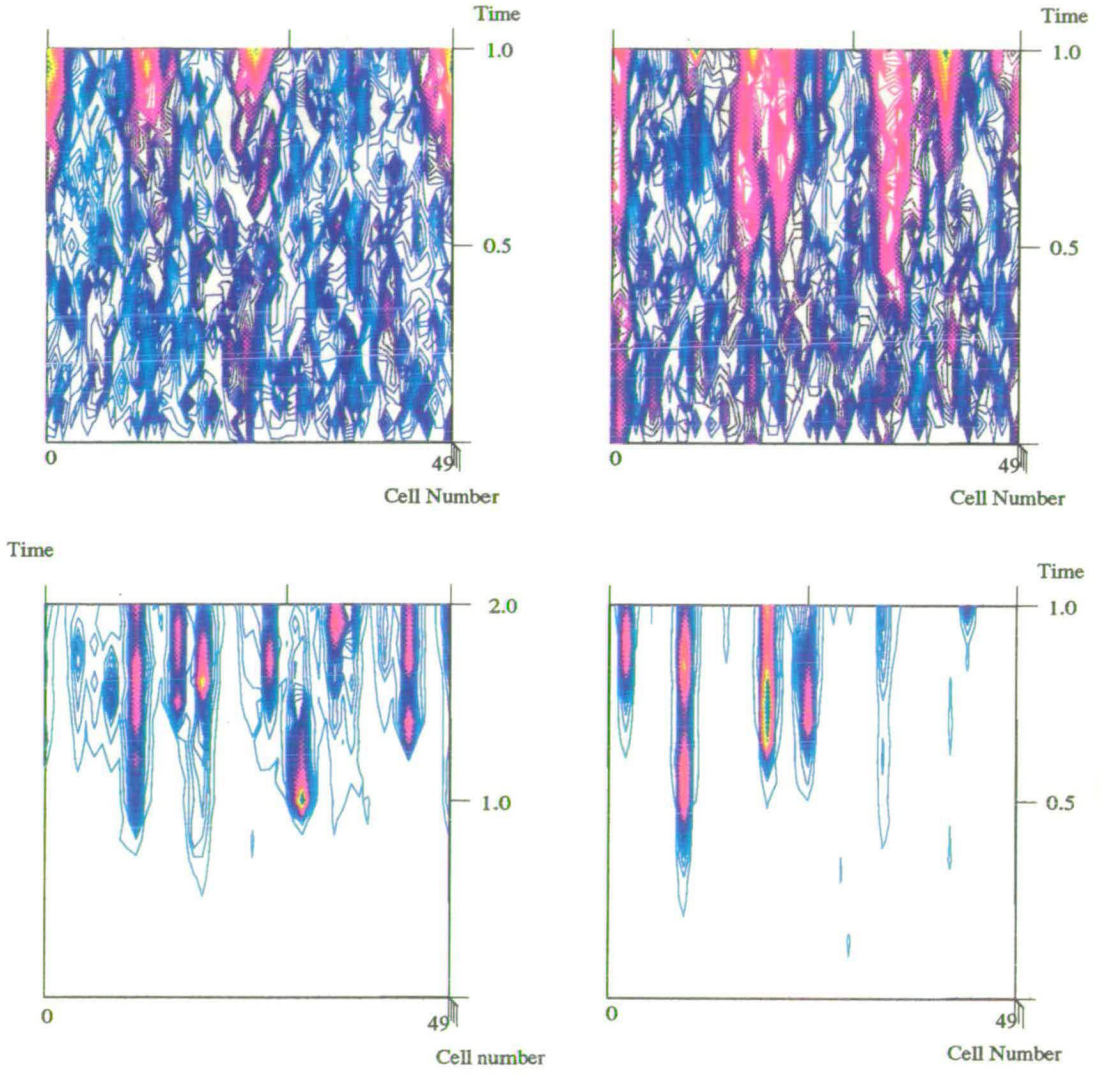

Figure 5.16: Variation of stochastic wave development with periodicity
These contour plots show the spatial and temporal development of stochastic wave structures in a one-dimensional spatial reaction system, following an initial equilibrium state. The system has non-linear super-critical ($I = 0.1$) interactions, and the four graphs represent the scenarios for a range of system periodicities. We would expect deterministic runs of these systems to produce two waves ($\lambda' = 0.04$, top left), three waves ($\lambda' = 0.06$, top right), five waves ($\lambda' = 0.1$, bottom left), and ten waves ($\lambda' = 0.2$, bottom right). It is clear from these graphs that reactant level inhomogeneity is increased with system periodicity. For low λ'-values populations appear fairly well balanced, however as λ' increases high population *hot-spots* become increasingly severe.

the performance related work is covered in Chapter 7.

Other two-dimensional investigations that are not covered in later chapters are those relating to the special cases of Turing's linearised spatial reaction system. In particular, we have simulated stochastic versions of the cases of "extreme-long" and "extreme-short" wavelength scenarios. These correspond to the limiting case activity in our standard linearised spatial reaction system.

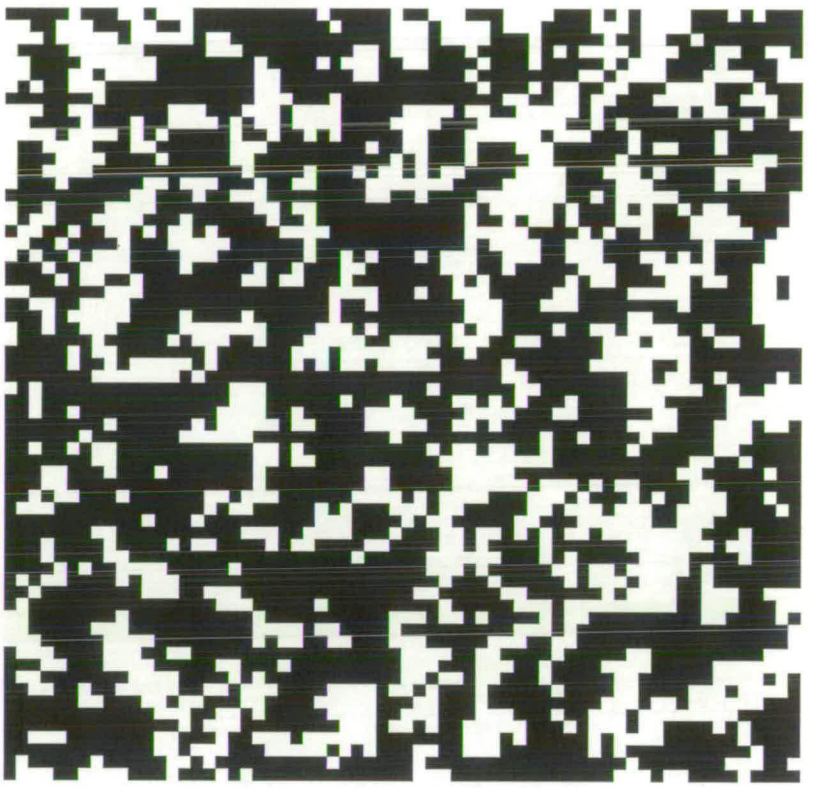

Figure 5.17: Extreme-long wavelength two-dimensional stochastic waves
Using Turing's [1952] parameters to produce extreme-long wavelength stochastic waves, we obtain the following graphical output from a two-dimensional simulation on a 64×64 location world. Each location is coloured black if the X-reactant population is below the equilibrium level X^*, and coloured white if the population $X \geq X^*$. Note the "patchiness" of the solution with neighbouring locations tending to contain the same class of population.

Figures 5.17 and 5.18 show the results of two-dimensional simulations of spatial reaction systems in which interaction parameters are set to reproduce the "extreme" wavelength behaviours discussed earlier in Chapter 2. These examples of graphical output show the clear tendency for "extreme-long wavelength" scenarios to produce neighbouring locations with similar reactant population levels; whereas "extreme-short wavelength" scenarios produce locations with neighbouring populations that are highly dissimilar.

Turing [1952] suggests that the former case could be related to the *dappled* patterns exhibited on the skins of many species of animal. This initial idea has since been investigated extensively in the deterministic domain (e.g. Murray [1989]), and is now ripe for further stochastic study on very large spatial domains.

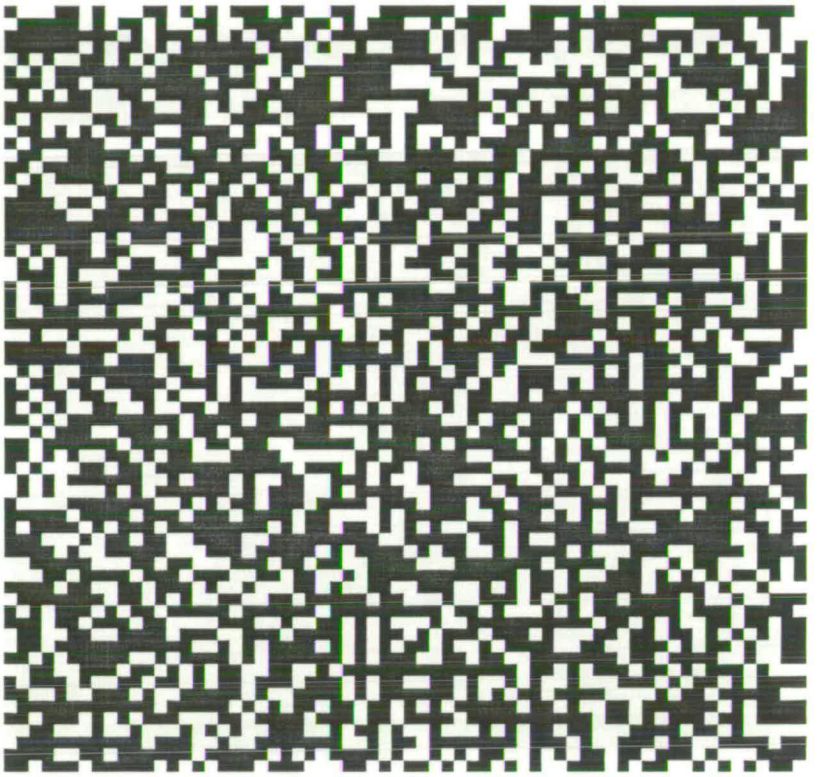

Figure 5.18: Extreme-short wavelength stochastic waves

Using Turing's [1952] parameters to produce extreme-short wavelength stochastic waves, we obtain the following graphical output from a two-dimensional simulation on a 64×64 location world. Each location is coloured black if the X-reactant population is below the equilibrium level X^*, and coloured white if the population $X \geq X^*$. Note the almost "checker-board" effect of the solution as neighbouring locations tending to contain the opposite class of population to their neighbours.

6

DDR — Dynamic Load Balancing on MIMD Systems

To isolate mathematics from the practical demands of the sciences is to invite the sterility of a cow shut away from the bulls.

– Pafnuty Lvovich Chebyshev

6.1 Introduction to Dynamic Load Balancing

Many scientific programs can be characterised as working with a set of particles or events that are free to interact and move within a *world* described by a regular or irregular mesh or grid. Such programs are usually parallelised by decomposing the grid into a number of sections and having one processor carry out calculations for each section. When this *spatial decomposition* technique is used, processors must synchronise when exchanging boundary values (see Fox *et al* [1988]) in order to ensure consistent iteration generations across the processors. This synchronisation comes automatically with SIMD systems, but must be enforced by the programmer in "loosely coupled" MIMD machines. If the work-loads of the processors differ significantly, than those with less work will frequently be idle, waiting for their neighbours' boundary values to arrive. Since these lightly-loaded processors could spend that idle time doing calculations on grid points assigned to heavily-loaded processors, it is clear that there is potential for improvement in the overall speed of the program through consideration of a different mesh or grid decomposition.

For many deterministic problems the work-load for any section of the grid can be determined *a priori*. Using this information, the programmer can decompose the grid so that each processor will have an approximately equal load. Alternatively, if work-loads are irregularly distributed, such that it proves difficult to create equally loaded contiguous sections of the grid, or if work-loads are not known in advance, then the programmer can break the grid into many small fragments and distribute these randomly among the processors (see Fox & Otto [1986]). This technique is known as *scattered spatial decomposition* (SSD), Nicol & Saltz [1990] have shown using the Central Limit Theorem this method can ensure that each processor receives a roughly equal amount of work.

However, if the amount of work required on a section of grid varies with time, both regular and scattered decompositions can be ineffective, though for different reasons. Regular decomposition suffers because increases in work-load may be localised, thereby increasing the loading on some of the processors. Employing SSD can avoid this if a sufficiently fine-grained decomposition of the mesh or grid is used. However, for architectures with relatively high communications costs we cannot decrease grain-size indefinitely, since our overall communications overhead will increase with the ratio of

patch boundary to area. In the worst case, with single-point patches, every point in the grid becomes a "boundary" point. We therefore find that many stochastic systems with time-varying work-loads can only be implemented effectively by using some form of *dynamic load balancing*, in which work is (re-)distributed at run time. This idea was probably first postulated by Stone [1977] during early work on computer networks, and is becoming ever more important for a wide class of scientific problems (see Hong *et al* [1988]).

Early suggestions of the need for dynamic load balancing were centred within the systems programming side of computer science. The work of Chow & Kohler [1979] provides a review of the concepts of *task farming* — the centralised distribution and scheduling of many independent tasks within a (potentially heterogeneous) computer network. While no estimate of the work-load associated with each part of the problem is available before execution, by decomposing the total work into small independent jobs, and distributing these jobs on demand, the programmer can ensure roughly equal loading of processors. A large body of later work has since concentrated on how to allocate tasks to processors within such environments: for example Hwang *et al* [1982] provide a *bidding* algorithm in which the currently smallest loaded processor receives new tasks; and the later work of Eager *et al* [1986b] concentrates on a comparison of "receiver-initiated" and "sender-initiated" load sharing policies for the migration of tasks to free processor resources. Unfortunately this latter work does not produce any conclusive evidence for either policy being preferable in the general case.

As research into dynamic load balancing progressed, work started to move onto ever larger computing systems, and it soon became apparent that for such systems a centralised task allocation mechanism was highly inefficient. This is due in most part to the bottleneck created as a single processor is used to output tasks and receive results. The work of Bryant & Finkel [1981] is an early attempt to counter this problem through the development of a task scheduling algorithm that is distributed across the available processors. Although a stable algorithm is produced, performance is seen to be severely hampered by the phenomenon of *thrashing*, which means that tasks are continually transferred between processors rather than undergoing execution. We will see this same problem occurs in early implementations of our own dynamic load balancing algorithm. However, a variety of mechanisms have been found to counter the thrashing problem, as well as many others that have hindered the development of a truly general mechanism for

dynamic load balancing of independent tasks. The initial work of Lin & Keller [1984] replaced a central queue of tasks with a distributed "balancer" that measures local work "pressure" and migrates tasks when the pressure gradient between processors is over a certain threshold. In a fine paper from a year later, Ni, Xu & Gendreau [1985] highlight a solution for thrashing, and also introduce some of the problems involved with how to identify processor load and how to deal with the transfer protocol information required to allow decentralised load balancing. Both of these latter problems become even more complex when considering calculations that cannot be considered as independent tasks. The methods used within our new decentralised re-mapping algorithm (DDR) to solve these problems will be detailed later. Shortly after the work of Ni, Xu & Gendreau this field had advanced to the state of the first paper classifying the various decentralised balancing strategies for independent tasks (see Wang & Morris [1985]), and in the following year Eager *et al* [1986a] produce the first analytic evidence that even simple dynamic load balancing strategies can yield high performance at low costs. Although there has been some recent interest in moving back towards the centralised approach for particularly well-suited problems (see Ahmad & Ghafoor's work on "semi-distributed" load balancing [1991]), it is now generally accepted that some form of distributed balancing algorithm is ideal for task farming on a large number of processors.

Most research in the past decade has concentrated on systems in which we cannot use the general simplification that the problem can be decomposed into independent tasks. This is certainly the case for our spatial reaction system, in which the localised activity at any instant in the simulation can have a direct effect on other locations. If we were to task farm this type of problem, our implementation could suffer from excessive communication costs and, more importantly, the interaction between neighbouring grid points would have to be considered by the central scheduling process at each iteration. What is needed is a combination of the two approaches, namely retaining connectivity of the mesh or grid, but re-mapping it as hot-spots develop and decay.

Although, a variety of approaches have been suggested to solve this problem, unfortunately none of them allow us to solve the general system represented by our spatial reaction system. Nicol & Saltz [1988] suggest that such "data parallel" applications require the use of a centralised *remapper* that can be invoked at a global synchronisation point in the execution; though no such point is available in general models. Nicol & Reynolds [1990] use an advanced, calculation intensive static re-mapping algorithm

that is dynamically invoked only when a recognisable phase-change occurs within a simulation; however, such systems prove very expensive if substantial load distribution changes occur at regular intervals. There is a school of enthusiasm (typified in the work of Boglaev [1992]) for the use of periodic exact global re-mapping calculations, which would again prove very expensive to implement for spatial reaction systems. In fact Hong *et al* [1988] stress that such an approach is only possible for a restricted class of problems.

In this chapter we describe an implementation of a dynamic load balancing technique called *Decentralised Dynamic Re-mapping* (DDR) tested on our standard stochastic spatial reaction system that has been analysed and simulated in earlier chapters. We have developed a mechanism that not only solves many of the problems highlighted above, but also provides very low computation-cost load balancing of a high quality. We find that DDR avoids the overheads of centralisation, and yet remains flexible enough to be applied to a variety of grid-based problems (see Smith & Wilson [1991]). DDR achieves near-perfect load balance for our general problem at a very low cost, and results in execution time improvements of over 40%.

6.1.1 The Model to be Load Balanced

Mesh- or grid-based simulation models are well established for use in numerical solutions of physical systems on a spatial domain. However, the mathematical techniques for analytic solution of such systems, particularly in the non-linear regime, are not currently so advanced. In the earlier chapters of this work we studied spatial reaction systems in some detail, and found that Turing's [1952] model can be used to analyse any spatial system where reactants/particles have a particular location within which they interact, and neighbouring locations to which they can migrate.

Our particular interest in spatial reaction systems is to enable a study of complex ecological systems, and in particular to study the effects of mutation and evolution. We therefore desire to implement complex stochastic simulations of interacting species, where certain attributes of each species can be successively adapted through random mutation. If these implementations are then sufficiently efficient, and simulations can be run of large systems over long time periods, then we should be able to present

results that describe the effects produced by mutation and evolution within the system. More importantly, the implementation techniques developed in this work have a much wider application than stochastic spatial reaction modelling. In our system we are working with a highly unbalanced data structure that requires some added work in order to distribute it evenly across a number of processors. The balancing techniques described in this and the next chapter are therefore applicable to a very wide range of numerical simulation work. For example, in multi-particle dynamics simulations we deal with large numbers of particles existing within some spatial domain – these may well cluster in certain regions; in meteorological modelling we are often confronted with highly uneven distributions of sensor locations, leading to a very unbalanced loading of a regular spatial grid; in cosmological modelling galaxial structures form in highly localised regions of activity; in Geographical Information Systems we may be studying point-locations which may again be clustered in certain small areas of our grid-based model. The strategies we are about to describe could be applied to any of these problems in order to support efficient parallel implementations. In fact, some of the key ideas from this work have been incorporated in general purpose parallelisation tools developed under the Parallel Utilities Library (PUL) Key Technology Programme at the Edinburgh Parallel Computing Centre (see Trewin [1992]).

To allow an analysis of the performance of our load balancing strategies for general case simulations, we introduce a generalisation for the local interaction process. We feel that the migration process is far more consistent between systems, as we are restricting ourselves to simple nearest-neighbour migrations of reactants. By comparison, the local interactions within different systems can vary substantially in terms of the computational effort required to compute each iteration. For example, in some epidemic models there may be a relatively simple interaction between a single infective and each susceptible in the location, whereas in cosmological modelling we may have to calculate complex all-to-all interactions between many elements. We have therefore introduced the idea of model *complexity* (C) as a measure of how much computation must be done for each reactant in each iteration. This measure is proportional to the number of floating point operations needed for each reactant. For our simple spatial two-reactant system this complexity value is taken to be unity. As the number of reactants increases, this value obviously rises linearly with the number of potential interactions within the system, as well as with the added work of extra model features such as evolutionary behaviour,

or environment dependence of interactions or migration. We will see from our results later (particularly in Chapter 7) that the performance of our dynamic load balancing strategies can be highly dependent upon the complexity of the system under simulation.

6.2 Implementation of DDR

Let us consider our one-dimensional spatial reaction system with periodic boundary conditions, in which reactants can migrate to neighbouring sites around a ring of locations. At each location we therefore have a certain set of reactant populations for which we must make calculations. We also assume that these cells make up the indivisible calculation unit size for the problem (i.e. a task that cannot be divided between processors for calculation), and that they may vary by orders of magnitude in work-load. This indivisibility assumption is required for two reasons. First, the stochastic nature of the simulations is produced by a stream of random numbers generated on a per-cell basis. Second, when we consider that each reactant's actions are dependent upon the current location population totals, if this is divided between processors, then severe communications patterns can become necessary to keep track of all events that may effect the evolution within a cell.

While such a system is a simple one-dimensional mesh, the load balancing strategies we have studied are intended to scale-up to be useful for more complex problems. Indeed, the DDR strategy could be improved, for example through the incorporation of problem specific information, for our one-dimensional spatial reaction system itself, but any such simplification would lead to a loss in generality.

The overriding design feature of a dynamic load balancing algorithm must be that the operations involved in calculating changes in processor responsibility, and in transferring work between processors, should not outweigh any efficiency gains achieved. Accordingly, we adhere to the following principles in designing the distributed dynamic re-mapping algorithm:

Local Decision-Making: Load balancing decisions must be made in a truly distributed fashion for a scalable solution, as shown by Eager *et al* [1986a]. All load balancing decisions must therefore be made locally, based on communications

only with near-neighbours. Although this restricts us to decisions based on partial knowledge, such sub-optimal balancing has been shown to yield good results at low costs (see Barah & Shiloh [1985]).

Local Balancing: For similar reasons, all transfers of work must be made to near-neighbours only, allowing work to diffuse from heavily-loaded to lightly-loaded processors. This helps to ensure contiguity of the mesh across processors, which minimises the latency of boundary value message traffic. This work-migration approach has proved very successful for independent tasks in a network (see the advanced work of Lin & Keller [1987]), and has been shown to converge to an optimal balance as a quadratic in the number of processors by Boillat [1990], a study that provides a dynamic load balancing system that is very similar to the initial DDR algorithm detailed later. We show that although complex heuristics must be developed, local balancing can be successfully employed for general grid or mesh codes.

Use of Existing Communications: Since processors must communicate regularly to exchange boundary data, load balancing information should be included in these messages, rather than as separate communications, as suggested by Ni, Xu & Gendreau [1985]. This will reduce the number of message start-up costs incurred — a significant saving on many systems.

Infrequency of Balancing: If it takes a significant time for load imbalance to develop or decay, then it would be wasteful to attempt to continually re-map data. It has been claimed by Williams [1991] that in such *quasi-dynamic* scenarios we can afford to redistribute the mesh based on global information. However, we prefer to maintain complete balancing flexibility by allowing the load balancing algorithm to make re-mapping decisions periodically and with variable frequency. It is in this area that our analytic knowledge from earlier chapters can be used to enable the most effective choice of how often we should attempt to re-map data — based on our statistical understanding of the occurrence and longevity of reactant hot-spots.

Our choice of MIMD load balancing strategy is obviously shaped by our target architecture, whether it be the original platform for this work — the Edinburgh Con-

current Supercomputer (ECS), a Meiko Computing Surface containing over 420 T800 transputers — or more recently produced machines that tend to be based around standard workstation (RISC) type processors. With such platforms processors can typically be reconfigured to any required topology. The DDR load balancing software has been developed in the C programming language, using Meiko's CS Tools [1990] programming environment, which includes a software router offering anywhere-to-anywhere connectivity. Our software uses this functionality to divide the available P processors into a single *master* process and $W = P - 1$ *worker* processes. The master handles user interaction and file I/O, while the workers carry out the simulation calculations.

DDR operates by adjusting the boundaries of responsibility between worker processors. Decisions on when balancing should take place are made locally by processors that control neighbouring sections of the mesh. To do so, they compare times taken to complete the last iteration. If the time taken by a worker process to complete one iteration is significantly greater than that of its neighbours, then the worker may give its lower boundary cell to its lower neighbour, or its upper boundary cell to its upper neighbour. The protocol for achieving this transfer is included in the boundary value messages normally passed between processors. Along with the number of migrant reactants, each worker includes the population of the appropriate boundary cells, and the time it took to complete the last iteration. Each worker can then compare its own time with those of its neighbours, and should a cell transfer be required it already has the necessary cell populations with which to work.

To permit flexibility in the load balancing strategy, three parameters were implemented to control the operation of the technique:

Time_Thresh: In certain circumstances it would be wasteful for neighbouring workers to transfer cells whenever there is any difference in their calculation times. Therefore to limit the amount of load balancing, a threshold λ_t was used. Thus a worker will pass a cell to one of its neighbours only if the difference between their respective timings for the last iteration is greater than λ_t.

Cell_Thresh: In order to ensure that no worker ever emptied itself of work, a second threshold parameter allowed workers to pass cells only if they had more than λ_c cells. This cell threshold was set to unity for many of our simulations, but can

be increased for large simulations in order to optimise performance or limit the amount of memory required on each worker.

DLB_Freq: Since location imbalances may change relatively slowly, it is not always necessary to attempt to load balance every iteration. Accordingly, a third parameter f_l, the load balancing frequency, is used to control how often load balancing is attempted. Since the DDR load balancing function is highly efficient in this implementation, very little is lost by re-mapping the problem more regularly than was necessary. However, for general applicability of the strategy to all types of MIMD computers, this parameter was included to enable optimal efficiency and future adaptability,

We can therefore define the DDR strategy for transferring a cell across a particular boundary by using the pseudo-code protocol detailed in Figure 6.1. This section of code is called following the receipt of boundary exchange information, but before local interaction calculations are begun. The interim counter records the number of iterations completed between each call to the load-balancing code.

```
IF    (Interim counter equals DLB_Freq) AND
      (I have more than Cell_Thresh cells) AND
      (This cell is my smallest edge cell) AND
      (My calc time - neighbours calc time > Time_Thresh)
THEN (Transfer Cell)

IF    (Interim counter equals DLB_Freq) AND
      (Neighbour has more than Cell_Thresh cells) AND
      (Neighbours edge cell is his smallest edge cell) AND
      (Neighbours calc time - My calc time > Time_Thresh)
THEN (Receive Cell)
```

Figure 6.1: DDR dynamic load transfer heuristic
This pseudo-code details the structure of the heuristic protocol used by each processor in order to decide when to transfer a cell of work to a neighbouring processor.

6.2.1 Initial Problem Decomposition

For simplicity we require the number of cells in the ring N to be an integer multiple of the number of workers W. Each worker j is therefore initially given $n = N/W$

contiguous cells, running from cell jn to cell $(j+1)n-1$, as shown in Figure 6.2. Each

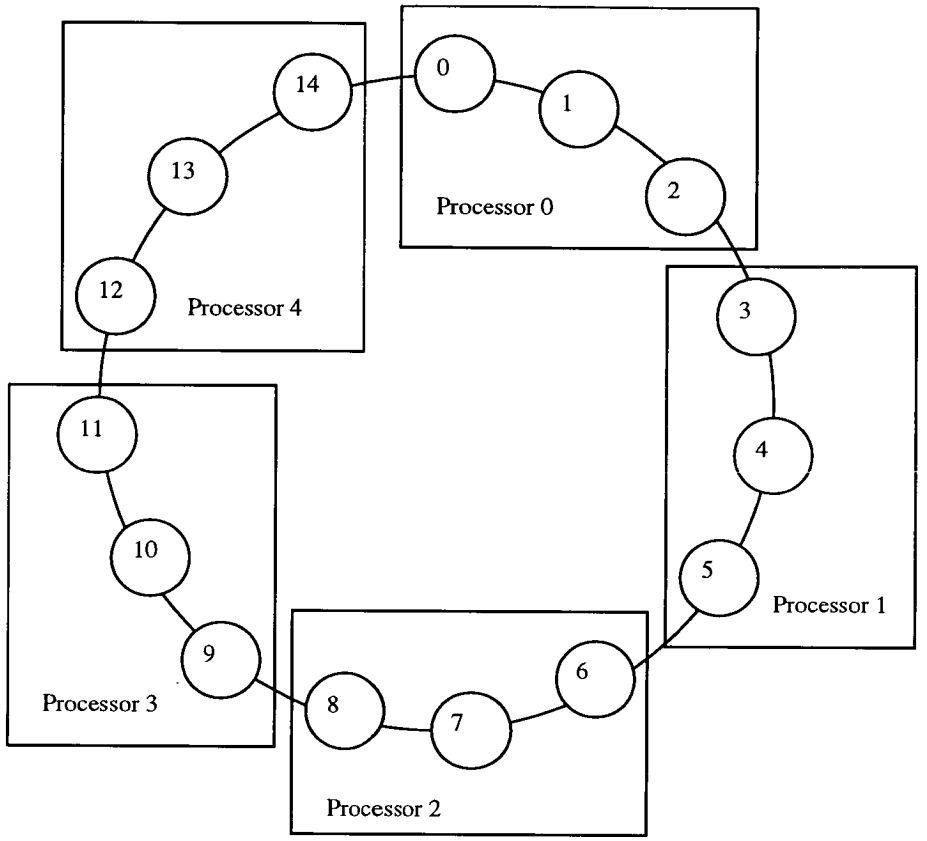

Figure 6.2: Initial DDR cell distribution

Simulation locations, numbered 0–14 in this example, are initially distributed evenly between available worker processors 0–4. Cells on each processor are maintained as a contiguous section of the one-dimensional mesh.

worker keeps track of its cells' populations in separate arrays for each reactant. For each worker, these arrays are made large enough to store the entire ring, in order to simplify the load balancing implementation. In fact, processors only need to allocate enough space for $N-(W-1)\lambda_c$ cells, where λ_c is the cell threshold described above. Each worker maintains an index *base* to show the start in the reactant arrays of the section it is responsible for, and a count *num* to indicate how many cells it has to update (see Figure 6.3).

During each time step of the simulation, each worker process calculates new reactant populations for its cells, and determines the number of reactants migrating in both directions around the ring. The time taken to do these calculations is directly proportional to the total population of the worker's cells. Workers then exchanged migration information with their ring neighbours, adding incoming migrant values to their boundary cells;

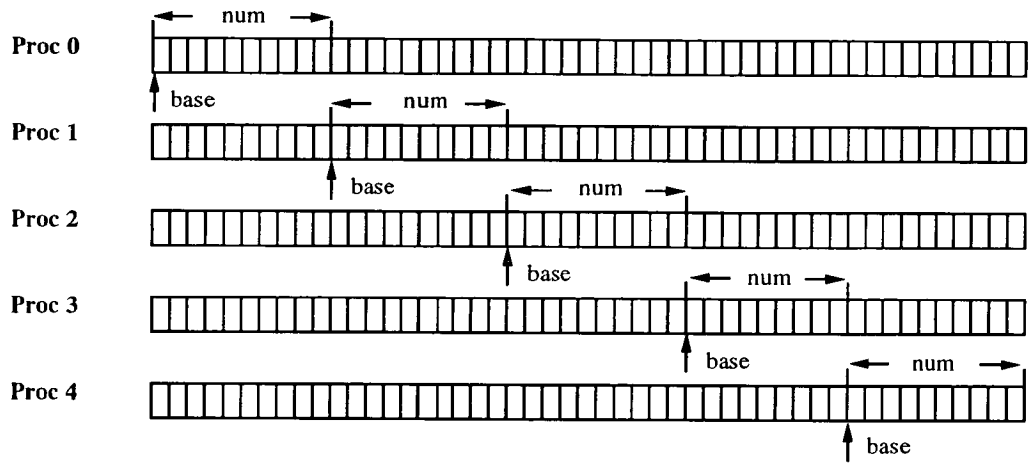

Figure 6.3: DDR's maintenance of array sections

Each processor holds arrays for each reactant, maintaining their current range of responsibility using local variables *base* and *num*.

the time required to do this is constant and independent of cell populations.

In the first version of DDR, each worker dealt with each of its boundaries independently. If there was a significant difference between the calculation times on either side of the boundary, then the processor with the lower time would accept responsibility for its neighbour's boundary cell, incrementing its `num` counter and, if the cell was arriving on the lower boundary, decrementing its `base` index to include the new cell in its update calculations (see Figure 6.4). Similarly, the processor with the higher time would decrement `num` (and possibly increment `base`) to exclude the transferred cell from its calculations. This strategy failed to load balance the problem efficiently because heavily loaded workers would continually pass small-valued cells to their neighbours until a high-valued cell reached the boundary. Neighbouring processors would then pass this high-valued cell back and forth, as each flipped between being lightly and heavily loaded, depending upon whether they held the large population cell. This *thrashing* was observed by Bryant & Finkel [1981] in some of the earliest work on distributed load balancing, and in our simulations it led to large oscillations in the calculation time on each worker, and thus to a high overall iteration time.

In order to overcome the problem of thrashing we developed an improved dynamic load balancing strategy. In this advanced version of DDR a worker only passes the boundary cell with the lower population if it finds that it is taking more than λ_t time units longer to complete an iteration than both of its neighbours. This encourages the

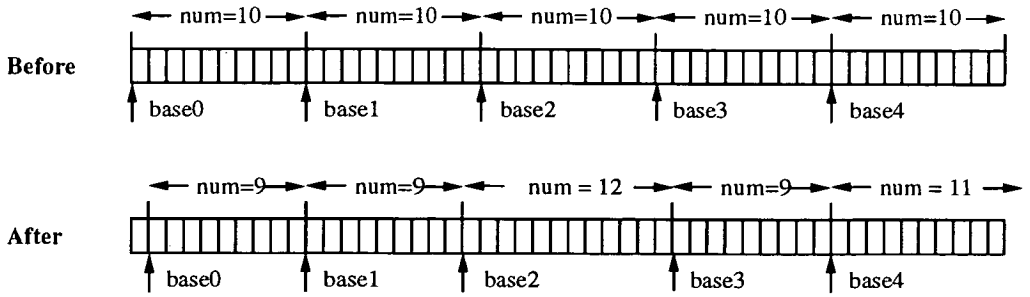

Figure 6.4: Updating boundaries in DDR

This diagram shows the mechanism for re-mapping work within the array of processors. When it is decided that responsibility for cells must be passed to neighbouring processors, the necessary reactant values are exchanged and recorded in the reactant arrays, and the processors involved simply adjust their local value for `base` and `num`. In this example we see processor 0 pass cell 0 to processor 4, and processors 1 and 3 both pass cells to processor 2.

retention of high-valued cells. In addition, if a worker's execution time is less than that of either neighbour it may accept either one or two cells, depending on which cells those neighbours are willing to transfer. Finally, if a worker's time is between that of its two neighbours, it may accept a cell from one, and pass a cell to the other. This effectively passes a cell around the ring, although this can also be thought of as a processor moving its domain of responsibility one step around the location array. Figure 6.5 shows these three alternative work transfer cases.

6.3 Dynamic Load Balancing Results

There are certain problems with claiming exact results when dealing with the simulation of stochastic systems. Since we use random number generators that reside on particular processors, when we make load balancing decisions that change the work particular processors must undertake we can change the very nature of the simulation. We must therefore collect statistics from a sizable number of executions, in order to gain any insight into the performance of our strategy. The timings that we detail later show quite clearly that the cost of using DDR amounts to a small proportion of run-time for models of all complexities. Although this result is due, to some extent, to the inherent efficiency of our use of the features of a particular message passing system (i.e. exploitation of the high ratio of message start-up to data transfer costs), it enables us to to make very good

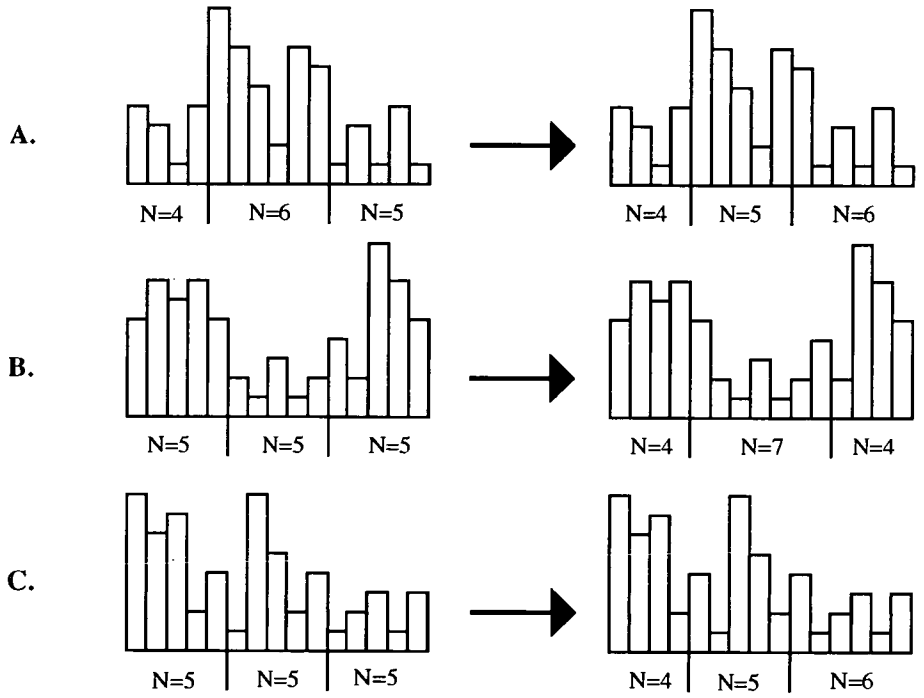

Figure 6.5: Boundary Cell Swapping Cases in DDR

This diagram shows the three alternative scenarios for dynamic load transfer within DDR. In case (A) the central processor only passes its smaller boundary cell (to the right), despite the fact that it is more heavily loaded than either neighbour processor. In case (B) the central processor accepts a new cell from each neighbour. In case (C) the central processor accepts one cell from the left and passes another to the right — thereby effectively moving its domain of responsibility one cell around the ring.

execution time savings.

6.3.1 Performance Prediction

In the general case we can regard the execution time of a perfectly balanced spatial reaction system simulation (without using DDR) as consisting of a proportion of time spent on calculation (αC) and a proportion for data communication ($a + bt$), where a is the start-up cost for the message passing, t is the amount of information in a message, and b is the cost to send each item of data. The additional cost of running DDR on such an implementation can be defined as the sum of the additional calculation cost DDR_c (the number of calculations involved in load balancing decision making) and the extra message passing cost $DDR_m t$, where DDR_m is the number of extra pieces of information that must be transferred to enable the DDR balancing protocol to operate. We can therefore state that the proportional run-time cost (Ω) of using DDR for a single

iteration is

$$\Omega = \frac{DDR_c + DDR_m t}{\alpha \mathcal{C} + a + (b + DDR_m)t} \quad .$$

We can assume that DDR_c will always be small, especially when compared to α. In fact, for our typical problem $DDR_c \simeq 5$, whereas α is typically of order 100–1000, and DDR_m is comparable to b. The overhead of using DDR is therefore most highly dependent upon the relative costs of the transfer of the balancing information ($DDR_m t$) to the unavoidable costs of the simulation calculations ($\alpha \mathcal{C}$) and the message passing start-up cost (a). High values of a and $\mathcal{C}$ will therefore give rise to low overhead balancing by DDR. We can also use the cost measure, Ω, to produce a relation to give the maximum possible cost (D_{MAX}) of using DDR (as a percentage of execution time) for any complexity and balancing frequency values, i.e.

$$D_{MAX} = \Omega / f_l \mathcal{C}.$$

When implemented on our particular hardware and software systems, for the case with the highest profile balancing — i.e. lowest complexity ($\mathcal{C} = 1$), highest frequency ($f_l = 1$), and artificially enforced perfect balance in the system — DDR has been empirically shown to be responsible for only 2% of execution time. This exceptionally low cost is mostly achieved from two factors. First, the relatively high start-up costs of our message-passing system (for CS Tools $a \simeq 200$ms, and $b \simeq 2$ms) enable the additional balancing information to be transferred at negligible additional communications cost. Second, our simple load balancing heuristics mean that DDR_c is small compared to α. This empirical result can be confirmed against the known parameters of our system, and hence a value of D_{MAX} can be determined for any further simulations, for given f_l.

Although the above relations show that DDR can be a very low-cost dynamic load balancing technique, is does not give us any real understanding of the success of the strategy in reducing overall simulation run-times. If we define the work-load associated with a cell n as L_n, then for N cells distributed to W workers the ideal balancing would provide all processors with a loading of

$$(1/W) \sum_{n=0}^{N-1} L_n.$$

The performance of DDR will always be limited by the relative size of the largest single work-packet ($max(L_n)$, say) compared to the best mean loading of the remaining work on the remaining processors. Given knowledge of the worst case largest indivisible work-packet, we can produce a *degradation factor* (χ) for the performance of a DDR-balanced application by giving a lower bound on the ratio of the best possible DDR balanced run-time to the best possible run-time. It can be shown that (assuming $\lambda_c = 1$) this degradation factor is given by

$$\chi = \frac{(W-1) \times max(L_n)}{\sum_{n=0}^{N-1} L_n - max(L_n)} \tag{6.1}$$

and so DDR will always fail to give optimal balancing if any one cell's work-load, $max(L_n)$, is greater than S/W where S is the sum of all cell work-loads.

We know from the explosive nature of super-critical spatial reaction systems as detailed in Chapter 5, that it is often the case that a single cell's population can dominate a particular stochastic simulation. Under such circumstances DDR will never be able to achieve perfect load balance due to our restriction of retaining complete cells on individual processors. However, we do find that the technique has been successful in producing execution times close to the best possible, even compared to computationally expensive post-processing to perform re-balancing using the *Knapsack* algorithm (see Sedgewick [1988]).

The results detailed in this section are for sub-critical simulations ($I = -0.2$) as these are naturally subdued and do not suffer from extremes of imbalance, or *singularities* in reactant populations. We concentrate on this class of system as results for super-critical simulations are inherently linked to the run-time for the single largest cell, and DDR rapidly adapts load distribution to solve this problem in all cases. However, such a solution may not be optimal if the single largest cell is dominating the calculation load (see Equation (6.1)), and thus some alternative load balancing technique may be more suitable than DDR. For example, the "parallel-prefix remapping" technique introduced in the following chapter may be a far better solution. Although extreme populations can also occur in spatial reaction systems at or below criticality, and thereby prevent perfect load balancing, sub-critical simulations do not generally suffer from a large amount of such behaviour. Therefore, by tackling this sub-critical scenario we are better able to analyse DDR's performance for a simulation in which active load balancing is required

regularly, and can actually produce a close to optimal implementation.

6.3.2 DDR Performance Results

Table 6.1 details the executions timings achieved over many sample runs of our one-dimensional spatial reaction system. The mean execution time from ten simulations, and their standard deviation, are recorded for each pair of parameters λ_t and f_l. Unfortunately, it is impossible to compare exactly one particular balanced run to an identical unbalanced one. As soon as any work is transferred between processors the random number generator will produce completely different series of events. It would have been possible to create generators that were specific to a cell, and therefore attempt to produce duplicate runs. However, this was unnecessary for two reasons. First, equivalence between balanced runs can never be guaranteed since the balancing decisions are dependent upon clock cycle figures. These figures can vary by ± 1 for identical calculations, and this can be enough to change the nature of an individual simulation. Second, it is not clear that taking an average result from a number of direct comparisons from particular runs is a more representative measure of performance than simply averaging a number of independent runs.

λ_t	100		80		60		40		20	
f_l	Time	σ_t	Time	σ_t	Time	σ_t	Time	σ_t	Time	σ_t
1	25.24	1.72	26.56	2.21	25.87	2.06	25.78	1.27	26.23	2.45
50	25.87	2.43	25.93	2.44	24.94	2.21	25.29	1.70	26.32	2.68
100	26.41	2.68	25.73	2.06	26.11	2.71	25.77	2.28	25.46	1.22
200	25.16	1.92	25.00	2.70	26.57	2.17	25.98	1.63	26.73	3.30
400	26.63	2.96	26.13	2.43	27.32	2.56	25.97	2.64	26.39	3.98
600	26.98	1.65	28.77	3.01	29.20	3.32	26.94	1.71	26.86	2.53
800	28.81	3.23	28.29	3.32	27.77	2.29	27.22	2.13	29.23	2.75
1000	29.10	3.43	30.21	2.71	28.46	2.64	30.23	3.71	30.20	2.50

Table 6.1: Distributed dynamic remapping (DDR) performance figures
Timings (in millions of processor clock cycles) for our standard spatial reaction simulations in one dimension using sub-critical non-linear interactions ($I = -0.2$). Dynamic load balancing parameters λ_t and f_l are varied with λ_c held constant at unity. The simulation uses 80 reaction cells on 16 T800 (transputer) processors.

For the particular size of model chosen the mean timing over many unbalanced runs was

33.15 million processor clock cycles (MegaTicks), where one clock cycle is $64\mu s$ (i.e. a "MegaTick" equates to 64 seconds). Figures 6.6 and 6.7 give a graphical representation of these results. It is observed from these graphs, and Table 6.1, that calculation time is reduced for every parameter set investigated. Even for high values of load balancing frequency there is a significant saving to be made; this is in part due to the slow changing nature of the sub-critical spatial reaction system model. For super-critical applications the choice of f_l could be far more critical as these systems can change their morphology far more rapidly. In addition, there seems to be little difference in execution time savings for different time thresholds (λ_t). Again this may be something that differs for other types of system.

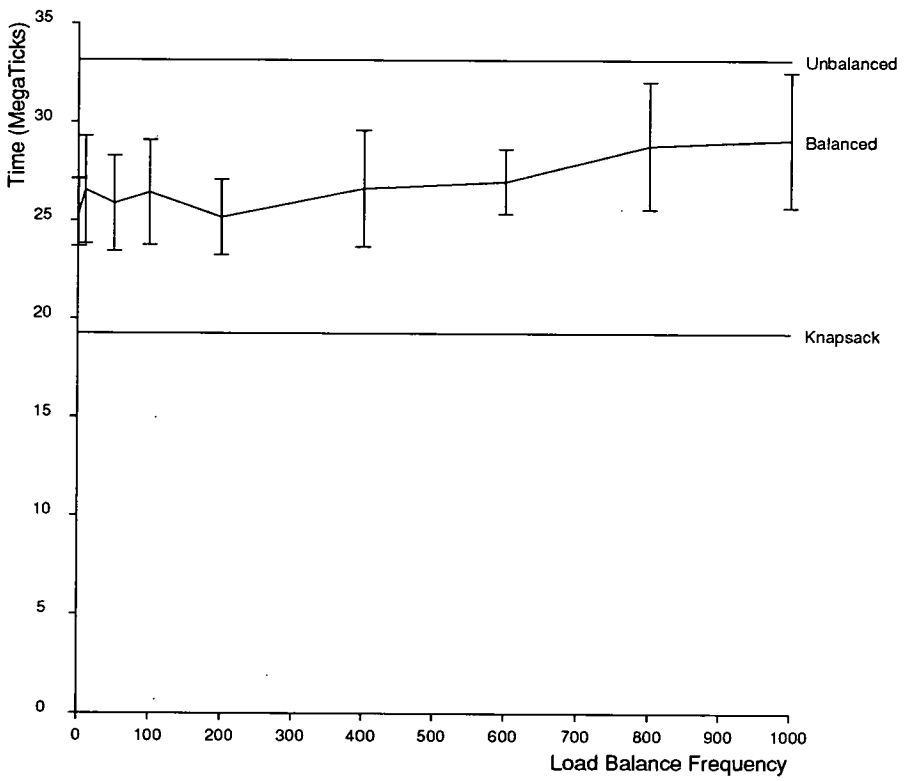

Figure 6.6: Load balanced execution times with low time threshold
Mean run times for our standard spatial reaction simulations in a one-dimensional sub-critical ($I = -0.2$) system, using DDR with $\lambda_t = 100$.

As mentioned earlier in this chapter, execution times for programs with and without the load-balancing code show that the cost of using DDR is very small. For this particular model approximately 2% of execution time is spent load-balancing in a simulation run when load-balancing is attempted at every iteration. As detailed above, this is achieved

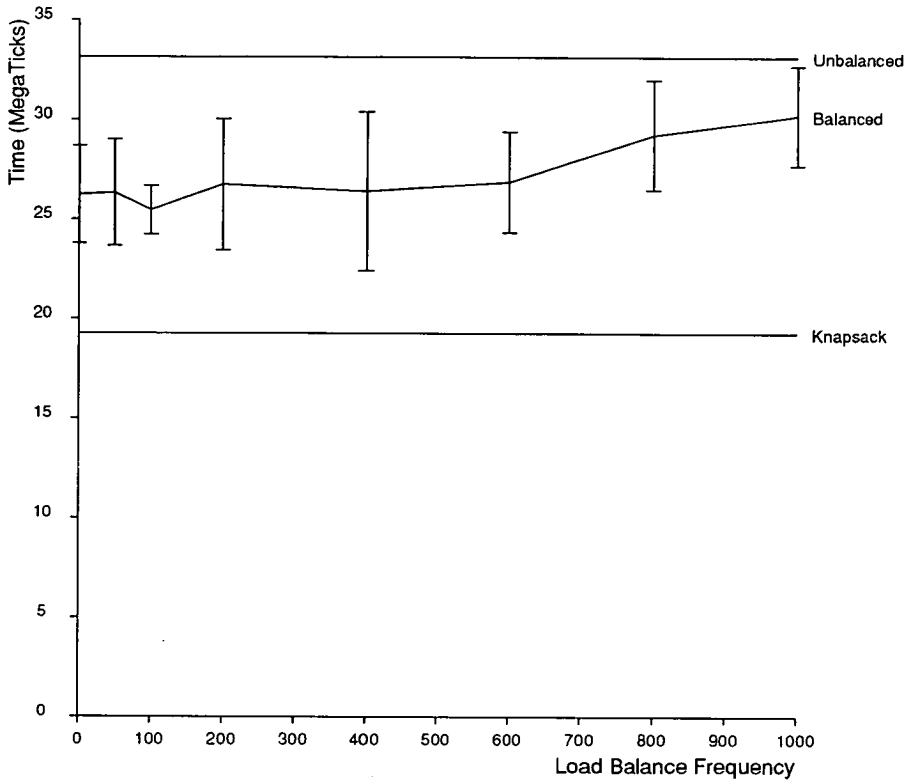

Figure 6.7: Load balanced execution times with high time threshold
Mean run times for our standard spatial reaction simulations in a one-dimensional sub-critical ($I = -0.2$) system, using DDR with $\lambda_t = 20$.

by the successful exploitation of existing communications traffic for message passing, and the simple rules that define when cells are transferred.

DDR reduces the average run times for sub-critical spatial reaction system simulations by up to 40%, a substantial saving for these inherently balanced (but very dynamic) systems. As mentioned earlier, for super-critical systems DDR results can almost always match the best possible result, as it soon reduces to hold the single largest cells on a single processor. From the start of this work the aim has been to design a load-balancing strategy that could be applied to grid- or mesh-based problems in general. It is therefore necessary to insist on processors controlling contiguous sections of the mesh, and for individual sites to remain indivisible. The graphs in Figures 6.6 and 6.7 show the mean reduction in run time that could be achieved for sub-critical systems by using the *Knapsack* algorithm to balance the simulation, assuming it incurred no execution costs (which is known to be false). The algorithm collects timing information for

each cell, sorts the cells by this timing, and then distributes them in order among the available processors, always placing the next cell on the processor with the least work. This Knapsack approach retains the indivisibility of cells, but ignores the contiguity of processor allocations. It also introduces large execution time costs not shown on these graphs (the program written to analyse data output by the simulations had a run time of several times the actual simulation run-time), which would only be increased further by the additional communications that would be necessary for implementation. The Knapsack line is therefore included on the graph to give some target for the absolute best possible speed-up given cell indivisibility, although not location contiguity. A numerical analysis to find the line corresponding to the additional restriction of contiguous sections on processors has not been attempted, but given that this is a constricting factor, this line cannot lie below that of the Knapsack line, and in most cases will lie above it.

We therefore believe that DDR is an effective dynamic load balancing technique for general spatial reaction systems, and that it can be applied straightforwardly to a wide variety of other grid-based calculations. While the book-keeping necessary to implement DDR on a multi-dimensional mesh would be significantly greater than that required for a ring, we believe that the gain in speed due to improved load balancing would outweigh these costs, in particular if the technique is further enhanced to incorporate asynchronous periodic balancing. In such a system processors can initiate a work transfer whenever loads become unbalanced, rather than waiting for the next "balancing iteration". The basic concepts of DDR, as well as these further ideas are currently being incorporated in automatic dynamic load balancing parallelisation libraries for both regular grid and unstructured mesh problems within the Parallel Utilities Library (PUL) project at the Edinburgh Parallel Computing Centre (see Trewin [1992]). These libraries are used for a variety of commercial projects to implement industrial strength software on high performance parallel computer systems.

7

PPR — Dynamic Load Balancing on SIMD Systems

Not only in research, but also in the everyday world of politics and economics, we would all be better off if more people realised that simple non-linear systems do not necessarily possess simple dynamical properties.

– Robert M. May

7.1 Data Parallel Load Balancing

In Chapter 6 we reported on the development of dynamic load balancing algorithms for one of the major classes of high performance parallel computers — MIMD machines. Much of our simulation work, however, has made use of the other class of parallel supercomputers — SIMD architecture machines. As discussed in Chapter 5, SIMD (or data parallel) machines can provide an extremely powerful computing resource, especially for specialised problems. During the course of this work we have had access to two different data parallel computing platforms, the Active Memory Technology DAP-608 and the Thinking Machines Corporation Connection Machine CM-200. Both of these machines were capable of providing the highest available levels of computing power available at the time of their use, provided that they could be programmed to make efficient use of the large numbers of processors that they contain. SIMD machines are typified by their massive number of processors; the DAP contained 4,096 processors, and the CM-200 contains 16,386 processors with 512 additional floating-point units.

The physical nature of the SIMD architecture, and the associated data parallel programming model, provides an ideal platform for the implementation of very large spatial simulations, as we can map our spatial data structures directly onto the spatial arrangement of processors. In particular, we have used these machines for our two-dimensional spatial reaction system studies. This approach has proved particularly productive, since the actual processor connectivity of SIMD architectures maps directly onto such two-dimensional grids. We detail in Table 7.1 the excellent performance results that we have achieved on these machines for deterministic realisations such as those detailed in Section 3.6. However, this chapter will focus mainly on new simulation algorithms developed to enable SIMD machines to be used for efficient stochastic simulations.

Since SIMD architecture machines obtain their performance by using massive numbers of very simple processors, they are more sensitive to processor load imbalance than similar powered MIMD machines. The reasons for this are three-fold: firstly, there are many more processors to be loaded with work, all of which must execute in complete synchrony, thus tasks requiring single calculations can delay all other processors; secondly, individual processors are typically low-powered and thus calculations not distributed across many processors will have a long execution time; and thirdly, due to the large number of processors, when few are performing useful calculation, this

configuaration will return only a very low rate of overall machine efficiency. Although these points suggest that effective load balancing on data parallel applications is vital when using such machines, there has been very little research into general balancing techniques. To a large extent this is due to the fact that the data parallel programming model is so restrictive as to make balancing functions such as DDR difficult to implement, and thus most SIMD machine users will adapt their implementation to ensure as balanced a load distribution as possible.

In this chapter we present a new algorithm for the efficient data-parallel implementation of stochastic spatial reaction systems based on individual reactants or particles. From our analytic work in Chapter 4 we know that such systems can be extremely unbalanced in terms of the number of reactants in different locations. We have therefore turned to a radical re-mapping of our implementation. Using this new approach (parallel-prefix re-mapping — PPR) we can ensure that all processors are evenly loaded, no matter how spatially imbalanced the reactants become or how dynamic reactant hot-spots may behave. However, PPR does introduce a computational overhead, and thus, if used on an inherently balanced simulation, could result in performance degradation when compared to a standard spatial decomposition. It therefore becomes vital to be able to predict the expected behaviour of spatial reaction simulations in order to decide on the best implementation strategy, as well as to estimate the expected simulation time. We must therefore use our knowledge of the mathematical analysis of the general stochastic differential equations that govern such models. This analysis can provide us with a statistical measure of system behaviour, which can then be used to predict the performance of both standard, and our new implementations.

7.1.1 Problem Definition

For the simulations studied in this chapter let us use our standard spatial reaction system in two dimensions. Thus $X_{i,j}$ and $Y_{i,j}$ (where $i, j = 1, \ldots, N$) describe the reactant population (or concentration) in each location (i, j) in a two-dimensional *world* of $N \times N$ sites. All our systems use periodic (i.e. toroidal) boundary conditions. In addition, if we let μ and ν be the rates of migration between neighbouring cells we can then write the following equations for the system:

$$dX_{i,j}/dt = \mathcal{F}(X_{i,j}, Y_{i,j}) + \mu(X_{i+1,j} + X_{i-1,j} - 4X_{i,j} + X_{i,j-1} + X_{i,j+1}) \quad \text{and}$$

$$dY_{i,j}/dt = \mathcal{G}(X_{i,j}, Y_{i,j}) + \nu(Y_{i+1,j} + Y_{i-1,j} - 4Y_{i,j} + Y_{i,j-1} + Y_{i,j+1}),$$

where $\mathcal{F}$ and $\mathcal{G}$ are the non-linear functions (see Equations (4.1) and (4.2)) that describe the interactive growth rates for reactants X and Y within each cell. Figure 7.1 shows the structure of this system in two dimensions, with migrations to and from nearest-neighbour locations. In earlier chapters we have detailed the analysis of this general type of system, based initially on the linearisation work of Turing [1952] in one dimension, and also on our stochastic representation of the one-dimensional system.

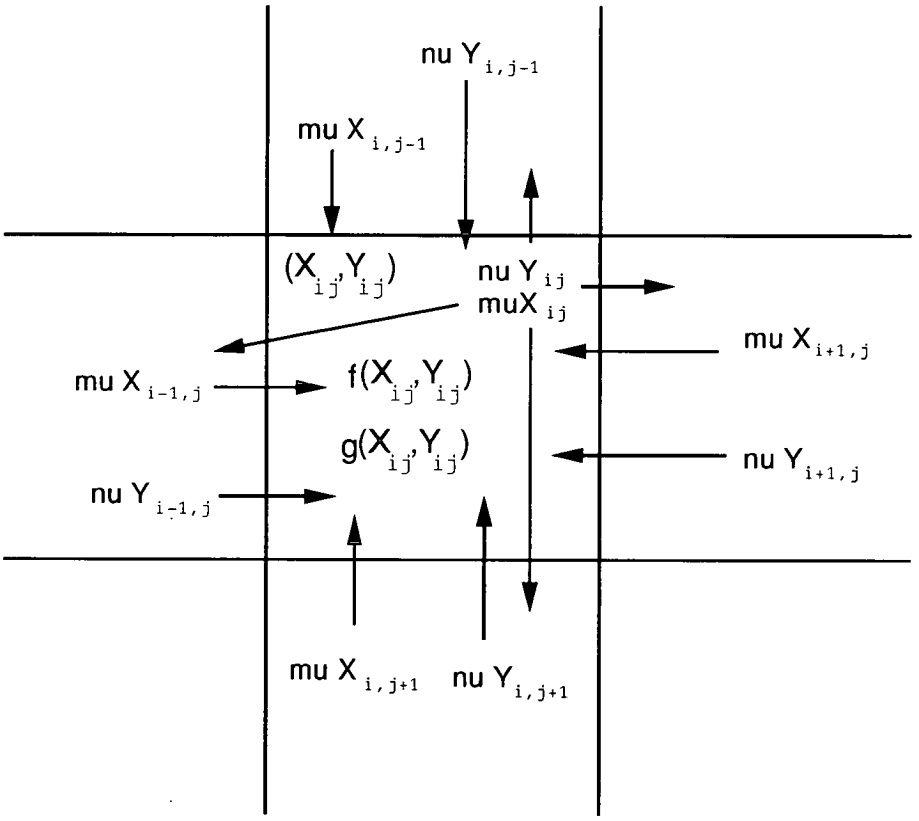

Figure 7.1: Nearest-neighbour migration options

The basic structure of spatial reaction systems in two dimensions, with nearest-neighbour migrations of reactant types X and Y occurring at individual rates mu and nu respectively.

This type of system has been used by Renshaw [1991] for a variety of one-dimensional stochastic simulations. First for a spatial Volterra [1926] predator-prey system, and

then for the case of stochastic wave simulations produced by using coefficients that produce inherently unstable local interactions. Our task now is to extend this work into two dimensions and implement numerical realisations of both the deterministic and stochastic forms of the system. For the case of systems that produce spatial and temporal waves we can follow the approach of Chapter 3 and define the eight interaction/migration coefficients in terms of just three system variables: the system instability I, the ratio of desired wave number W to world dimension size N, and the size of the equilibrium populations $X^* = Y^*$. The first and last of these parameters are calculated from the linearised values from our general non-linear interaction functions. For the latter we typically approximate the reactant migration rates to those from the one-dimensional analysis performed in Chapter 2. We see from the realisation and simulation results in Chapters 3 and 5 that this approximation proves remarkably accurate if we view the number of waves present along any axis within the spatial domain.

7.2 Spatial Decomposition Techniques

The most obvious and straightforward way of decomposing the spatial models described above is to perform a spatial decomposition. Such an approach is also obvious for other spatial applications such as molecular dynamics, lattice gas cellular automata and cosmological modelling. For data parallel implementations we can therefore construct an array of spatial locations (the array having the same dimensionality as the physical world we wish to model) which can then contain the value of the local populations in each cell — be that individuals, particles or galaxies. This array can then be distributed across the available processors within a SIMD machine. Local interactions, which may depend on the size of local populations, can be performed in parallel by the processor assigned to each array element. In addition, since migration is allowed only to nearest-neighbour locations, it can be performed using the inherent local communication facilities provided by the parallel architecture. Thus on the Connection Machine one could use either the `CSHIFT` or `PSHIFT` Connection Machine Fortran (CMF) functions for these operations.

7.2.1 Deterministic System Performance

This spatial decomposition approach works extremely well for systems for which there is an identical amount of work to be done for each location in the model, and hence for each array element within the machine. This is because the calculations are perfectly balanced across the machine, and therefore make maximum use of the available resources.

Machine	Implementation	CPU time (s)
Sun 4/20	Fortran 77	1615.2
CM-200 (8k)	raw CMF code	5.07
CM-200 (8k)	PSHIFT	4.09
CM-200 (8k)	optimised	3.83

Table 7.1: A comparison of data parallel execution times
Execution times are detailed for a deterministic realisation of our standard deterministic spatial reaction system realisation, using a 256×256 world and running for 1000 iterations. Timings are given for the original Unix workstation implementation, and three successive implementations running on half the processors (8,192) in the CM-200.

The results shown in Table 7.1 are for the deterministic realisation of a simple two-species spatial reaction system. This implementation uses two distributed arrays, each containing a single real number value that represents the current reactant population for each location. These values are updated, and migrating reactants exchanged, at every iteration. The "raw CMF code" version represents code that was ported from a Unix workstation implementation onto the CM-200 with minimum effort (a few hours work, admittedly by an experienced data parallel programmer, for 300 lines of Fortran 77). This transfer immediately produced a version of the software which ran in just over five seconds on half the Edinburgh Connection Machine. This performance was then improved by a further 19.3% through the use of poly-shift (PSHIFT) multi-dimensional communication routines, specifically provided within CMF to perform regular shifts to all four nearest-neighbours simultaneously. Finally, a further 6.4% run-time improvement resulted from other optimisations such as "code-blocking" — placing parallel code in contiguous sections within the program. We thus obtain a very impressive 844 times speed-up by running on the full Edinburgh CM-200 (16k processors) when compared to running on a desk-top workstation. Access to this level of simulation performance provides a new range of opportunities to the research

scientist. In addition to the ability to investigate a far wider range of complex systems, or particular systems to a far greater depth, this type of high performance solution provides access to methods of system investigation that were previously beyond the scope of computational feasibility. For example, we could now consider the development of an interactive simulation system in which parameters are adapted at run-time, and the effect viewed immediately, thereby allowing further exploratory variation.

7.2.2 Stochastic Simulations

As discussed in Chapter 4, although deterministic scenarios have proved adequate for modelling a great many physical systems, there is now a growing trend towards stochastic modelling for real-world simulations. Unfortunately, for stochastic simulations we cannot always guarantee that calculations are well-balanced across our distributed arrays, and when this situation arises we can easily make very inefficient use of parallel machines with a SIMD architecture. This results from the fact that all processors must perform identical operations in complete synchronisation, so all processors must perform the same number of operations as the most heavily loaded processor, even if they have much less work to complete. These unwanted operations are prevented from affecting simulation results by *masking* techniques (effectively turning off sets of processors for particular calculations), although they obviously do affect simulation times as the technique involves inefficient use of all processors. Our field of simulation study exposes this load-balancing problem particularly well, and we will shortly concentrate on our particular system. However, the same problems can be found in many other applications typically thought of as "difficult" to implement in a data parallel fashion.

Efficiency problems arise in data parallel implementations of spatial reaction system models as soon as we move from deterministic to stochastic simulations. There is much debate in the ecological and biological communities as to the relative merits of the two approaches (see Chapter 1 of Renshaw [1991] for a brief review). This has led to two main modelling groups; one that follows the ideas of May [1986] in that complex ecological behaviour can be explained by the fine structure found in deterministic chaotic systems; and the other is typified in the work of den Boer [1981] where nature is regarded

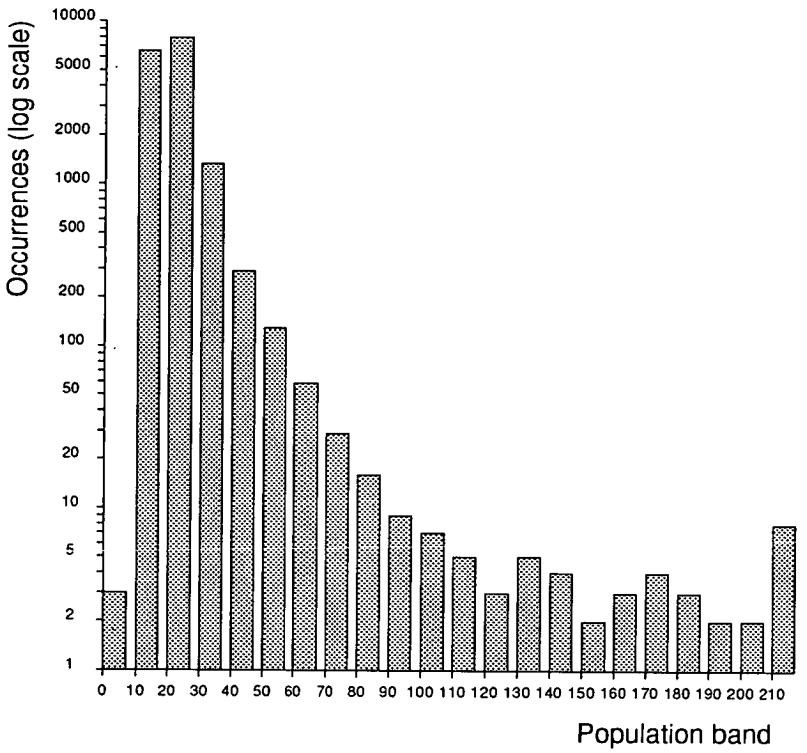

Figure 7.2: Histogram of reactant populations

A snap-shot population distribution histogram for reactant X in a 128×128 location spatial reaction system with $W = 8$, $I = -0.1$ and $X^* = Y^* = 10$. The final bar in the histogram represents the sum of all locations with reactant populations over 210.

as inherently stochastic and should be modelled as such. Since it is our ultimate desire to model mutation and hence evolution (both of which we believe to be natural stochastic processes), we are ourselves in favour of the stochastic approach.

As discussed earlier in Chapter 5, we immediately face load-imbalance problems once our models contain non-homogeneous population levels. This occurs because we need to make a set of probability calculations for each individual in the system, rather than a single calculation for a total population in a particular location. This involves the generation of many uniformly-distributed random numbers to compare against calculated probabilities, and is therefore computationally expensive on any computer architecture. Although one could approximate with a single stochastic calculation for the whole population of each location (hence balancing the implementation), for our evolutionary system studies we must track individuals, since each is potentially unique in terms of its current attributes. In this situation we are therefore forced to make separate probability calculations for each individual reactant.

If the stochastic system under study is highly stable, and hence contains populations

that never stray far from equilibrium positions, any slight imbalance between processing elements can probably be suffered. However, since we generally use inherently unstable interactions to produce the spatial and temporal wave patterns in which we are interested, we find that some cell populations inevitably move a great distance away from equilibrium. Figure 7.2 shows a typical population distribution histogram from a stochastic spatial reaction system (with negative instability, i.e. a fairly stable case) after just one time unit — 1000 iterations.

It should be noted that there is a logarithmic scale for the number of occurrences in Figure 7.2. We therefore see that in over 6,500 locations the reactant population is below 10, and in almost 8,000 locations it lies between 10 and 20. The number of occurrences then decays rapidly for higher populations. We find that there are two cells (from a total of 16,384) with a population between 200 and 210, and there is a total of just eight cells with a population of over 210. Figure 7.3 shows the formation and stabilisation of a typical population distribution from a set of these histogram graphs. Following an early narrow distribution as populations diverge from initial equilibrium, we get many population "explosions" that create a broad distribution by time $t = 1.5$. The later graphs show that the system then settles into a generally steady state, but with occasional high-population cells.

Over the course of a full simulation we find that the maximum cell population (X^M, say) will fluctuate between 150 and 800 for $I = -0.1$ (the range becomes 250-1500 for $I = 0.0$), and never more than a few cells will contain a reactant population with a magnitude close to this maximum value. It is therefore obvious that a simple spatial decomposition will be highly inefficient, since X^M random numbers will be generated, and X^M probability calculations will be made for every cell at every iteration, even though the vast majority of locations contain fewer than 20 individual reactants. The effect that this inefficiency has upon simulation execution times can be seen in Figure 7.4. This graph details the time taken to complete sets of 100 iterations on a 128×128 world, as well as showing the total number of individual reactants in the system. The upper line on the graph shows this total population, and it can be seen that this rises (in a relatively short time) from the initial configuration to a relatively constant level of just over one million individuals. The two lower lines in Figure 7.4 show the time taken to compute successive blocks of 100 iterations. The lower of these is for an artificial test case where the total population at each iteration (as taken from the actual simulation)

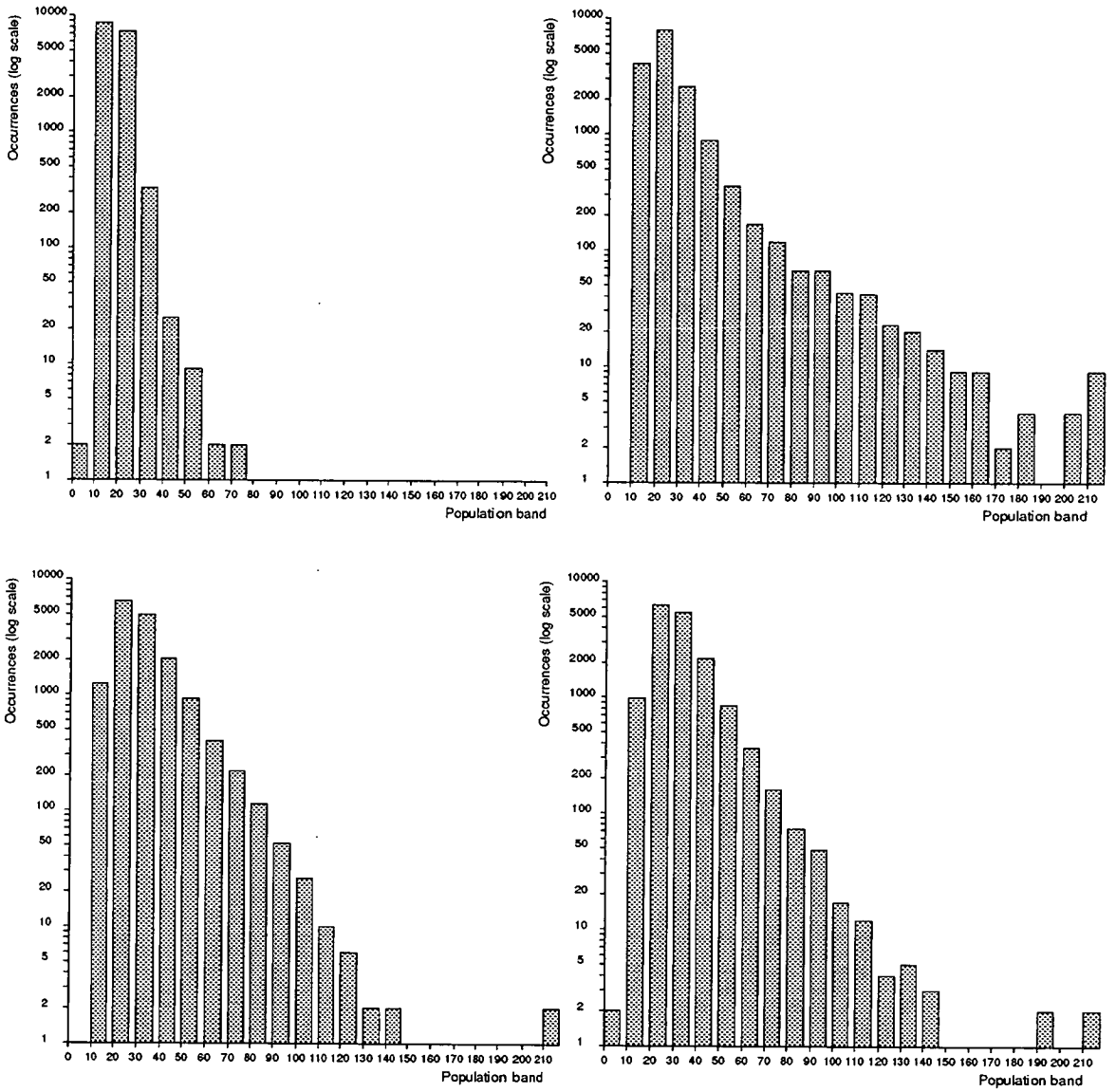

Figure 7.3: Reactant population distribution development

These four population histograms show the distribution of reactant X in a 128×128 location spatial reaction system with $W = 8$, $I = -0.1$ and $X^* = Y^* = 10$, at simulation times $t = 0.5$ (top left), $t = 1.5$ (top right), $t = 3.0$ (bottom left) and $t = 14.4$ (bottom right). The final bar in the histograms represent the sum of all locations with reactant populations over 210. It can be seen that the extreme imbalance in reactant populations grows rapidly, but then settles to a constant form, although the actual cells falling into each category will be constantly varying.

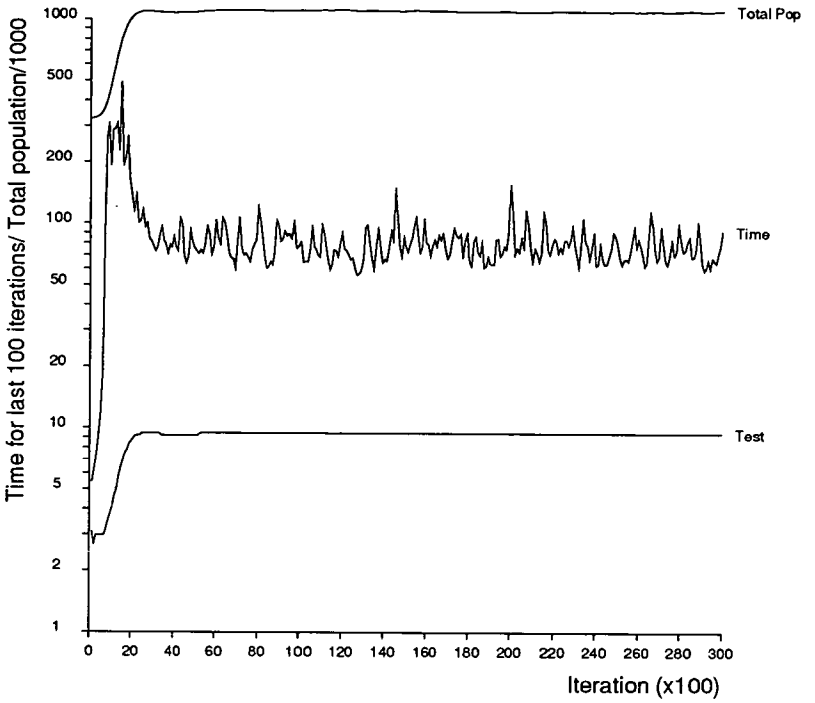

Figure 7.4: Stochastic populations and naïve execution times
Total populations and incremental run-times for a stochastic spatial reaction system with $N = 128$, $W = 8$, $I = 0.0$ and $X^* = Y^* = 10$. This graph shows both total reactant population levels (the upper line, Total Pop) divided by 1000, and timings taken for executing blocks of 100 iterations using both the standard spatial decomposition Time and an artificial perfectly balanced case Test.

is automatically distributed evenly across all processors. Interaction and migration probabilities are forced to be zero, thereby providing ideal balance but removing all reality from the simulation. However, we can use these timings as a comparison to the actual timings taken for a particular stochastic run (the central line). We observe the sudden growth in run-times as *hot-spots* emerge in the world, and certain cell populations grow rapidly. After approximately 400 time units the run-time for the last 100 iterations does settle down, but it is still (in this case with $I = 0.0$) around a factor of ten greater than for the test case. Indeed, in the early stages of the simulation the straightforward implementation can take over 100 times longer than the test case to perform 100 iterations. It is this level of discrepancy that we need to overcome if we intend to study systems on large spatial scales, at high complexities, or over very long time periods.

It can be noted from Figure 7.4 that the run-time of successive iteration sets is rather unpredictable for the unbalanced implementation, since it depends upon the latest value

of X^M. Also, it should be remembered that these timings are already an aggregate over 100 iteration steps, so the variation within such blocks could be even more severe. However, if we could evenly distribute this work, the execution times could become very regular, since the total population has reached its quasi-equilibrium value.

7.3 Data Decomposition Techniques

The obvious solution to the load imbalance problems of our spatial reaction systems is to consider another form of data decomposition. Since we cannot rely upon our locations to have equal work-loads, we must find another unit of decomposition that is balanced. In the case of our spatial reaction systems we can think of distributing the individual reactants among the available processors, since for each member of each species we have the same simple probabilistic calculations to perform. This will give us a very large data structure to work with (simulations often contain a million or more individuals, each described by a few integer values) but one that is evenly loaded in terms of calculations. The large number of elements in such an array should also ensure a relatively even spread amongst the many processors within a SIMD machine.

Unfortunately, such a change in data structure presents the programmer with two additional problems.

- It is generally the case that in order to perform local interaction calculations information must be available about other reactants in the same (or even neighbouring) locations. This type of information is easily available when using a spatial decomposition, but now it will be distributed throughout our reactant-based data structure. Such information can be contained within the local data structure that each individual *carries*, however this must be updated at each iteration and could be affected by events at any other location within the data structure (as reactants migrate). Such events will almost certainly be non-local to the processor, and we must therefore engineer the PPR software to keep efficient track of cell populations as well as individuals, and to allow mutual communication between both sets of information.

- We must allow our new data structure to be flexible, since the total number of

particles in our system will almost certainly vary over time, with individuals appearing and disappearing at random. We therefore need an efficient technique for retaining a compact data structure, with as few unused elements as possible.

In converting our simulation algorithm to this individual-based mapping we have converted the problem of a simple implementation with poor load balance, to a more complex (but balanced) algorithm potentially requiring substantial amounts of internal communication. Providing this additional communication cost is restricted to a reasonable level, we can obtain an implementation with relatively short, and also fairly predictable, execution time.

7.3.1 Parallel-Prefix Re-mapping

In order to present the parallel-prefix re-mapping algorithm, we must first detail the reactant-based data structure it uses. Figure 7.5 shows this structure diagrammatically for a two-reactant scenario. The array has a single *parallel* axis (running through the reactants, distributing them across processors), and a *serial* axis (data maintained within a single processor) for storage of data pertinent to each individual. This serial data is made up of both permanent reactant details, and temporary information used by the PPR algorithm. Figure 7.5 also details two logical arrays `Cflag` and `Sflag`; these contain logical flags to denote the cells that contain the first reactant of a new cell and type, respectively. These logical arrays therefore delimit the array into location and reactant-type sections.

The complete PPR algorithm is given in pseudo-code later in Figure 7.6. Before describing the main functions of its operation, let us concentrate on the methodologies used to solve the two specific implementation problems detailed above. To achieve their solution we make extensive use of parallel-prefix (or "scan") functions. Scans cover a range of operations that perform some combination calculation on all elements along a dimension of a parallel array; i.e. they compute for each element in an array the combination of itself and all previous elements on that particular dimension. Thus *add-scans* will sum all elements along a given dimension, and *copy-scans* will spread one value throughout an array. Parallel-prefix operations can also be *segmented*, whereby operations are executed upon separate sub-sections of the array, as delimited by some

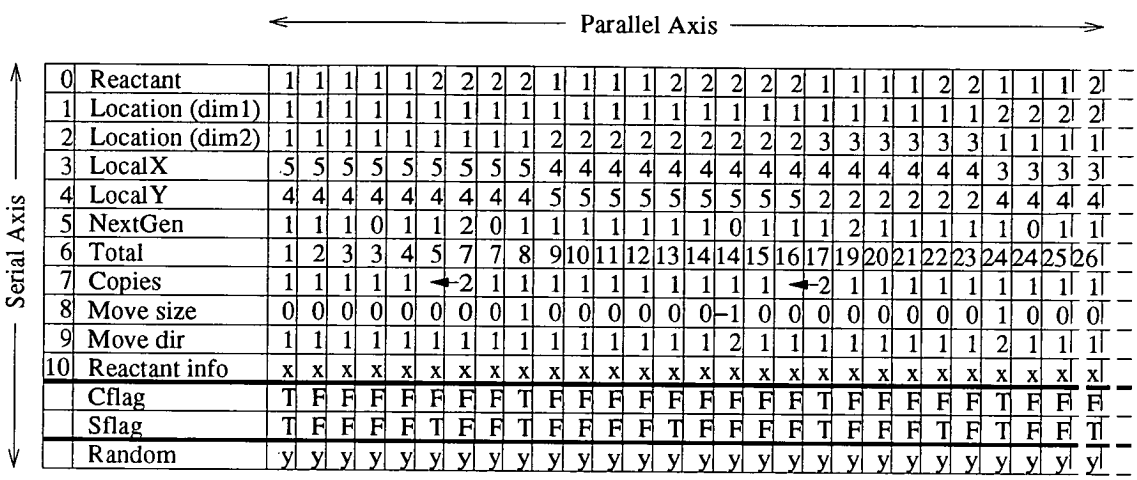

Figure 7.5: Parallel-prefix re-mapping array structure

A simple representation of the array structure necessary for the PPR strategy. Individual reactants are distributed along the *parallel* axis, and information pertinent to each individual is stored locally along the *serial* axis.

logical *flags*.

Obtaining local site information: If the reactant array is sorted with respect to the grid-location of each individual, then we can ensure that those reactants sharing the same location will occur in contiguous sections of the array. Then, by use of segmented parallel-prefix add-scans we can sum populations delimited by the flags given in `Cflag` and `Sflag`. These location-specific values can then be distributed to reactants using similarly segmented parallel-prefix copy-scans. If information is also required for neighbouring grid-locations, this can be achieved through additional parallel-prefix operations to pass sum values to neighbouring segments. However, transfer of such neighbour information can only be performed one dimension at a time, and thus involve a re-sort for each additional dimension after the first.

Retaining array contiguity: In order to create maximum efficiency of memory usage and program performance, we must ensure that all parallel array elements are occupied with data relevant to a "live" reactant. Thus, when one reactant leaves the simulation, or when a new reactant is created, the data that describes ot must be removed or added to the data structure in a manner that keeps the array contiguous. A mechanism for retaining this type of contiguity has been developed independently by both Boghosian *et al* [1991] and Nicol [1992], for

use with simulations with no specific location-dependence. However, we find that this technique can be directly adapted for this part of our algorithm, and it has the added advantage of retaining the existing order of the array. The technique uses a global parallel-prefix add-scan on the `NextGen` element in the reactant array. This provides a `Total` value that can then be used as the parallel index to which all serial data must be moved. This will *shuffle down* reactants to fill free array locations, as well as leaving space for duplications of reactants that have multiplied. These duplications can then be shifted down the parallel axis to complete the re-mapping. Should the simulation be in a reactant growth phase, we can use the final value from the `Total` array to indicate any need for dynamic array allocation, since normally, for maximum efficiency, the size of the parallel dimension will be kept close to the number of *live* reactants.

7.3.2 The PPR Algorithm

Based upon the two implementation techniques above, the main iterative loop of a PPR program (as detailed in Figure 7.6) moves the system forward in time, and undergoes the following operations for each iteration.

1. **Interaction calculations:** Using an individual uniformly-distributed random number and the location populations, each individual reactant makes a probabilistic calculation to decide whether it lives, dies or duplicates. The result of this calculation is stored in the `NextGen` element of the reactant data structure. There are only three possible values to be stored here; a zero if the individual has died, a one if there has been no change, and a two should the individual have duplicated.

2. **Re-contiguise:** This unusual term refers to the action of reorganising the distributed array following the changes brought about in the interaction phase. Such re-organisation activity involves the removal of gaps created by deaths, and the addition of new individuals following births. In order to create maximum efficiency of memory usage and performance, this reorganisation should be performed so that all gaps are removed and we have a single contiguous section of individuals

in the array. This can be achieved using parallel-prefix operations as detailed above, and the resulting values stored in the `total` element.

We can then use the values in the `total` element as the identifiers for the locations to which the contents of every currently live individual (i.e. `NextGen` equals one or two) should be sent. This allows the array to shrink or expand with the total population, with all gaps being removed. The final step of this process is to do a local copy for all the birth cases into the free space left beside them (see Figure 7.5).

3. **Migration calculation:** More random numbers are used to allow calculations to be made to determine whether each current individual undergoes migration. By comparing the number generated to the defined probabilities we can determine into which of the four possible directions an individual is to move. Here we use two more local elements of the array to describe any such movement. `Move size` gives the amount of change to be added to the current x or y location value, i.e. $+1, -1$ or zero. This change is then added to the value stored in the element that is indexed by the value stored in `Move dir`, i.e. either '1' or '2', defining x and y movement respectively. Using this slightly convoluted structure allows all migrations to be performed in one simple step, regardless of species type or direction of motion.

4. **Resorting:** The unfortunate effect of the migration of individuals is that the data structure may no longer be sorted according to location values. This occurs as the data will be sorted according to one dimension (say x) in priority to the other. Thus a migration in the y-dimension will produce an array element with a location value that no longer fits with the current sorted pattern. This order is necessary later when we wish to find new total cell populations, and thus the array must be resorted. There are sorting routines provided for the Connection Machine and these are used to sort the array in terms of x-location, y-location and then species.

5. **Retotalling:** Following the sorting process we must re-process the logical `Cflag` and `Sflag` arrays to find the location of new cells and species. These arrays can then be incorporated in add- and copy-scan routines to provide each individual with the latest value of species populations for its location. These routines perform a segmented add-scan through the array of individuals, summing members of each

species. This sum is reset each time Sflag is set to TRUE. Once these sums are known they can be copied to all reactants within locations by segmented copy-scans between the locations where Cflag is TRUE. By reversing the direction of this scan, we can achieve this *spreading* of totals using just two copy-scans.

```
Set-up initial array structure and probability tables Probs
Loop through time evolution of the simulation
      Local interaction calculation phase:
            Generate random numbers in Random
            Compare Random to Probs and add changes (±1) to NextGen
      Array contiguity phase:
            Global add-scan of NextGen into Total
            For elements 0-5 and 10
                  If NextGen.GT.0 send element to array position given by Total
                  If NextGen.EQ.2 copy-shift element down parallel axis
                  Reset empty elements, and NextGen to 1
      Migration calculation phase:
            Generate random numbers in Random
            Compare Random to Probs and set changes (±1) in Move_size
            and set Move_dir to dimension number
            Add (Mod N) of Move_size to value in element Move_dir
      Collecting local information phase:
            Sort parallel array by location and type
            Set Sflag TRUE when Reactant (i) different from (i − 1)
            Set Cflag TRUE when location changes between neighbours
            Downward add-scan of NextGen into LocalX, segmented by Sflag
                  masked by Reactant.EQ.1
            Upward add-scan of NextGen into LocalY, segmented by Sflag
                  masked by Reactant.EQ.2
            Upward copy-scan of LocalX into LocalX, segmented by Cflag
            Downward copy-scan of LocalY into LocalY, segmented by Cflag
next time step
```

Figure 7.6: Pseudo-code for the PPR algorithm

The complete PPR algorithm for a two-reactant spatial reaction system. The steps involved in the execution of this algorithm are described in detail in the preceding text.

7.3.3 The Performance of Parallel-Prefix Re-mapping

The parallel-prefix re-mapping strategy described above, and detailed by Smith [1993], works very successfully in terms of maintaining low and constant run-times for stochastic simulations. In particular we see from Table 7.2 that run-times for high complexity simulations can be reduced by up to factor of 17 over a simulation that is already running on the most powerful supercomputer in the UK. It does, however, suffer from fairly high overheads, mostly due to the sorting and scanning operations on such a large array. The cost of these operations is constant for constant array sizes, and thus the relative cost of using this technique is highly dependent upon the amount of calculation necessary for each reactant (i.e. the interaction complexity $\mathcal{C}$, introduced in Chapter 6) as well as the average imbalance of the location populations. For the simplest systems (complexity $\mathcal{C} = 1$) where each reactant undergoes two floating-point operations, the time to sort the reactant array (on the CM-200) is around seven times that for the interaction calculations themselves. When the time taken for the parallel-prefix operations is included, the result is an overhead of approximately a factor of eight over a perfectly balanced spatially-decomposed case. We therefore need a load-imbalance degradation of more than this value to warrant using PPR in this scenario.

However, since the costs of sort and scan routines are constant with array size, their relative overhead decreases as the system complexity, and potential imbalance, increases. This effect is clearly detailed in Table 7.2, as the run-time for PPR implementations becomes substantially lower than a variety of naïvely implemented simulations. The same table also shows the relative gains over systems with varying instability, where we can clearly see the effect of increasing imbalance (hence longer run-times) as system instability increases. It is also interesting to note the increase in variation of the execution times for the critical scenario. Although the overall run-time is below that of the super-critical case, we see a marked increase in the variability of the run-times. The increase in naïve execution time for super-critical systems is as we would expect from the analysis introduced in Smith & Renshaw [1993], and detailed earlier in Chapter 4.

Figure 7.7 shows graphically the effect of system complexity on the performance results for PPR and naïve simulations.

Type	Instability	Complexity			
		1	5	10	20
Naïve	0.01	14.3 ± 2.0	41.6 ± 6.0	148.0 ± 22.0	306.0 ± 45.0
	0.0	12.9 ± 3.8	37.4 ± 11.2	133.0 ± 40.0	275.0 ± 80.0
	-0.1	10.7 ± 3.0	30.7 ± 8.0	111.0 ± 31.0	230.0 ± 65.0
PPR		14.41	15.06	15.87	17.48

Table 7.2: Parallel-prefix re-mapping (PPR) performance results
Comparison of run-times for 100 iterations of a stochastic spatial reaction simulation. All timings are for a 32×32 world on 8,192 processors of a CM-200.

7.4 Future Improvements to PPR

The complexity measure used for the PPR results above represents the number of stochastic operations to be performed for each reactant. In the case of simple two-species ecological models this value may well be unity, and under such circumstances the use of the PPR dynamic load-balancing neither gains nor loses over the use of raw unbalanced code (other than for the obviously lighter implementation cost of the latter). It may be possible to develop other strategies for such simple models, and some ideas for these are covered in Section 7.4.1. However, as has been stated earlier, one of our main interests for the future lies in producing realistic models of ecological systems that exhibit evolution. For this reason our interactions will become increasingly complex, as will the environmental factors that must also be calculated at each iteration. Indeed, a surprisingly small step in model structure that lead to a system complexity that gains substantially from the use of PPR.

We also believe that the PPR technique could be useful for many other applications, such as molecular dynamics or N-body problems, where the calculations to be made for each individual can also be very large. Any unbalanced spatial reaction system in which reactants undergo more than one stochastic operation per iteration can gain from the use of our parallel-prefix re-mapping strategy. In fact, the more complex the interaction becomes, the more compute time (proportionally) can be saved, or perhaps more importantly, longer simulations can be executed in a given amount of user-time. In addition to this saving in run-time, PPR also provides the added benefit of consistent and predictable run-times. This in itself is a very useful feature when running many large simulations in order to collect results.

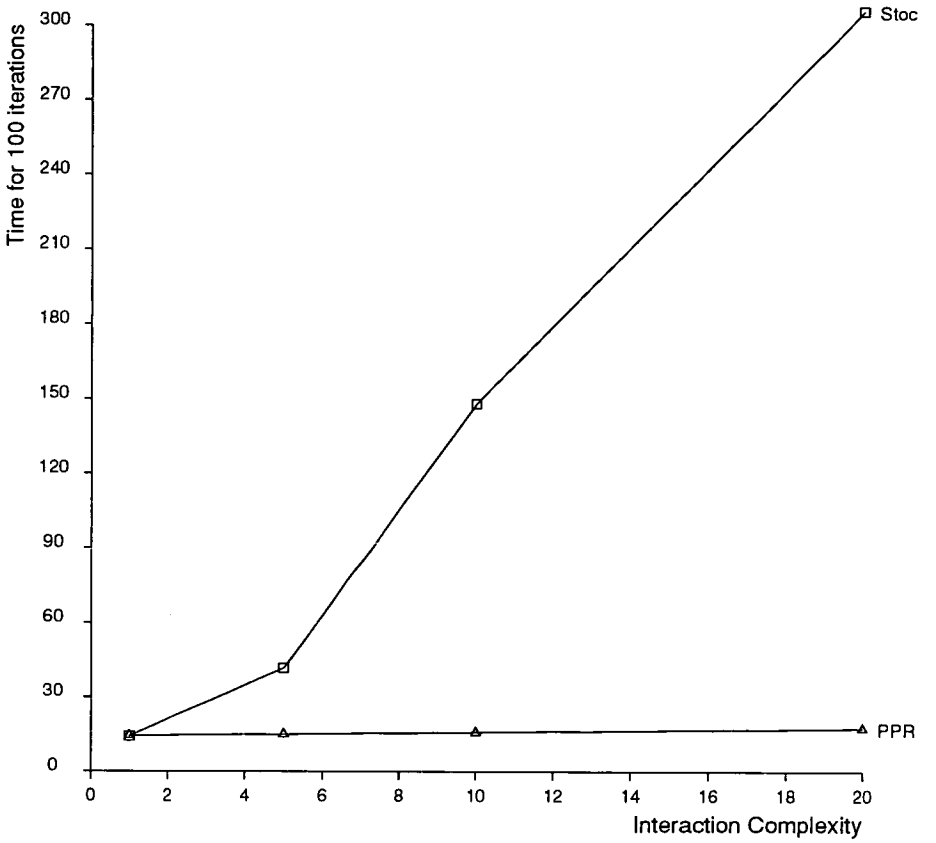

Figure 7.7: Performance of PPR and unbalanced implementations
Comparison of run-times for 100 iterations for different interaction complexities, between the load-balanced implementation (PPR) and a typical set of 100 iterations taken from the unbalanced runs (Stoc). These timings are for a 32×32 world, run on 8,192 processors of the Edinburgh CM-200.

7.4.1 Alternative Array Distribution Schemes

We believe that is is possible to develop a strategy that is a hybrid of PPR and the standard spatial decomposition technique, which maintains the balance and speed of using a distributed array of individuals as in the PPR strategy, yet avoids the costly sorting operations. This technique would use two separate array structures, both distributed across the machine, but with no direct relation between their distribution schemes. One array contains the individual members of each reactant-type, the other is a two-dimensional array that maps directly onto the model *world* and contains the populations of each location. By using this double array approach we would expect to retain the benefits of having easy access to location populations (hence removing the need for sorting), and be able to balance the calculations involved with the members of each species at the same time.

This planned strategy of double array distribution currently has three separate potential varieties. In the first, at each iteration all individuals would access the populations array for the relevant cell population data. They would also update this array following any events such as birth, death or migration. This strategy therefore replaces the resorting and re-totalling sections of the PPR technique with distinct communications between two distributed arrays. The parallel-prefix operations for maintaining the contiguous nature of the animals array is retained. The obvious problem with this first variant of the strategy is that when certain locations become hot-spots and have very large populations, many elements of the reactant array may be attempting to access the same single element in the population array. The bottleneck this may cause within the Connection Machines' internal communications network is difficult to predict accurately, but we expect that it would certainly hinder performance. To reduce any such effects we propose two further variants:

Record changes: We reduce the number of attempts per unit time to access the population array because it now records only changes in populations. If we can set logical flags in the reactant array to identify all individuals possibly affected by a change in cell population (probably using parallel-prefix operations), only these elements need to initiate any data transfer. In simulations that are only developing slowly, we would expect the majority of cells to not require updating in any single iteration. This may result in the saving of some communication costs.

Randomise population array: We retain the original double array structure, and introduce a third dimension to the population array randomised across the processors. The array access events can therefore be spread more evenly across the available processors. This would be the ideal solution giving us balanced individual calculations, direct access to population data, and no communication bottlenecks.

7.4.2 PPR Conclusions

We believe that the work we have detailed in this chapter has substantial relevance for real world simulation applications, which typically involve systems of high complexity and calculable instability. Our main intended application is in the field of evolutionary ecological simulation (see Chapter 8), where (like in many other fields) we have high

complexity models, and need to investigate systems with a wide range of instabilities. The main benefit to our work is the ability to make accurate execution time predictions.

We have presented a variety of mechanisms to aid the dynamic load-balancing of individual-based simulations. These techniques are very successful in balancing workload across the processor arrays of SIMD machines. Their usefulness depends upon the ratio of the amount of calculation involved with each individual in the system to the overhead of performing the balancing. When this ratio is large the PPR technique can provide substantial run-time savings, and even for models with small calculation levels per reactant (i.e. $\mathcal{C} = 1$) the technique provides run-times equivalent to a naïve implementation, but with much greater predictability. Work in this field is still progressing, and we have indicated the directions in which we are looking in order to provide a successful, but computationally inexpensive, mechanism for balancing these spatial individual-based problems.

8

Predator-Prey Studies with Evolution — A Supercomputing Case Study

If nature were not beautiful, it would not be worth knowing, and if nature were not worth knowing, life would not be worth living.

– Henri Poincaré

8.1 Introduction to Evolutionary Ecological Studies

In Darwin's [1859] "Theory of Natural Selection", evolution is driven by genetic mutations which occur, for example, through damage to DNA by cosmic rays. Although most of the resulting mutations are less suited than the original to their environment and therefore do not survive, occasionally a mutated individual may be better suited and thus grow in number eventually replace the original. It is thought that natural selection works by selecting characteristics which enable the species as a whole to survive. However, in Dawkins' recent books [1976, 1986] we are presented with an alternative viewpoint. Dawkins believes that Darwin's theory makes more sense if natural selection selects traits which enable the *individual* to better survive. He views the gene as the element which wishes to survive and reproduce as prolifically as possible; the species then becomes simply a means for the gene to reproduce.

As discussed in Chapter 1, the fundamental ideas behind evolutionary processes are gaining popularity in many disciplines. Unfortunately, any mathematical understanding of such processes is very limited, and thus the majority of studies attempt some computer simulation of "evolution" in order to investigate such matters. In this chapter we detail the results achieved through the use of parallel supercomputers for evolutionary ecological modelling. Such high performance computers have allowed us to simulate systems that are of a size and complexity thought previously impractical for computer simulation. In order to model true evolutionary behaviour, any mutation events must have a low occurrence rate, and must make only small changes to individuals. It is clear that we require simulations of many individuals, running over many iterations, if we hope to reproduce such evolution-like results. We therefore feel it is vital to use very large systems in order to claim any accuracy in our analysis of evolutionary effects. We report on the studies of Smith [1991] in the development of spatial predator-prey systems in which individuals migrate according to a set of behavioural rules. We then extend these rules to include mutation of individual's attributes. Our long time-scale simulations of large spatial systems then reveal how evolutionary pressure can produce individuals with attributes "selected" by evolution.

For seventy years scientists have been using mathematical techniques to study the dynamics of animal populations (e.g. see Lotka [1925] and Volterra [1926]). In more recent times concern has been expressed by den Boer [1981] at the reality of "universal determ-

inistic models". He showed that, in a heterogeneous environment, dispersion of prey is necessary to ensure the survival of isolated populations. In addition, Hilborn [1979] shows that a spatial predator-prey model that includes migration can provide a globally stable system from inherently unstable local interactions. Similar persistence of species has also been shown by Zeigler [1977] for stochastic spatial models where, in isolated cells, populations would rapidly decay to extinction. Zeigler studied the parameter region that provides global stability for a 100-site world, and states that this region is increased for a larger 900-site model.

Within the spatial model described in this chapter, predators and prey are able to migrate to neighbouring locations in a two-dimensional world. Dubois [1975] introduced a non-linear deterministic model of simple predator-prey relations coupled with diffusion to explain observed patchiness in plankton populations. However, that work and that of Levin & Segel [1976] and Murray [1975] have all found that deterministic patchiness or waves in populations soon decay to leave a "quasi-equilibrium" state. The stochastic study detailed here extends basic diffusion into preferential migration by both species according to a selection of behavioural rules. This successfully produces dynamic patchiness throughout our simulations, which can be related directly to the inhomogeneous population distributions observed in nature.

Previous ecological work by Shiyom [1980] accredited predators with certain levels of *mobility* and *attack ability*, in order to study the effects on the spatial patterns of a stationary prey. In this work we introduce some measure of power into the predators' attack ability, and allow the prey species to migrate, potentially according to preferential rules. In addition to studying the geographical distributions of the two species during simulations, we also compare the effect of different migration rules on the global stability of the system, and finally study the effects produced by the mutation of predator migration rates.

8.1.1 Supercomputers in Ecological Modelling

As we strive to increase the reality of ecological models by increasing the size of our model worlds, and by studying longer term effects in these worlds, we are often halted by the limitations of the computer doing the simulations. Onstad [1988] predicted

that supercomputers would be used increasingly for ecological study since theoretical models cannot be generally accepted if they are not realistic, and realism requires vast amounts of compute-time. There is increasing evidence (see Haefner [1991]) that Onstad's prediction is being realised, and that parallel computing technology is a major contributor to this advance. Haefner's review of this subject concentrates upon individual-based models and the various parallel architecture machines that are available, giving some references to current simulation work. In Conrad and Rizki's review of their modelling work [1989] there is regular allusion to the short-comings of available computer hardware. As their models became more complex, so the ability to study them in depth was reduced due to computing resource limitations. As detailed in Chapter 5, present semi-conductor technology is reaching the absolute limits set by the fundamental laws of physics; and we have therefore almost reached the maximum calculation speed possible from a single processor. The only mechanism left to increase computational power further is to multiply the number of processors being used. This fact is being acknowledged by every major computer manufacturer as they all look toward parallel processing for their future machines, see Trew & Wilson [1991].

As detailed in Chapters 6 and 7, potential performance gains from using a parallel computer are not always easily exploitable. The programmer must often expend extra effort to identify potential parallelism within a problem, and then convert this into writing a parallel program for a particular machine. Parallelism does not change the fundamental nature of the problem solution, it simply provides the means to solve larger problems in shorter times. The rewards provided by parallelisation can therefore be high — in the work covered in this chapter, large spatial models that would take days to study using a normal departmental mainframe computer were viewed (via direct high-resolution graphical output) and analysed in a matter of minutes. Our early work and results on preferential migration have all been obtained from the use of a DAP-608 machine. The more recent work on evolutionary effects exploited the Thinking Machines Corporation Connection Machine CM-200, and the dynamic load balancing technique (PPR) detailed in Chapter 7. Both of these machines are SIMD supercomputers, technical details of which were discussed earlier in Section 5.2.1.

8.2 The Predator-Prey Models Investigated

The predator-prey models studied in the course of this initial investigation are all chosen to make the maximum possible use of the parallel architecture of a particular SIMD supercomputer. The models have a spatial dimension in which each of the processing units in the machine takes responsibility for a section of the system. It is most straightforward to consider each processor as controlling a location that contains a certain number of each species. Thus for the DAP-608 for example, there are 4,096 locations in each model, arranged on a two-dimensional (64×64) grid. Periodic boundary conditions applied to the grid then allow boundary effects to be eliminated. However, cyclic effects are introduced, all be it on a large scale, and the model can therefore be likened to a stochastic Turing [1952] system in two dimensions. As the simulation evolves, calculations are made to decide whether an individual predator or prey will migrate, die or reproduce. This is therefore an individual-based model, in some ways similar to those studied by Zeigler and Conrad, although on a much larger scale both spatially and temporally.

8.2.1 Lotka–Volterra and Volterra Oscillations

The work of Lotka and Volterra in the 1920s laid much of the foundation for present models of population dynamics. The coupled differential equations they developed produce their now-famous deterministic population oscillations. The initial objective of our ecological work is to recreate these oscillations in a stochastic model that tracks all animals individually.

The model we use owes much to the work of Wolff [1989] in its design and parameter values. It is based upon predators having two states – *hungry* or *satisfied*, the state assigned depending on whether the predator has or has not made a kill in the previous time-step. The state of a predator then determines the probability that it may reproduce or die during that time-step. Additionally there are probabilities to determine whether members of either species will migrate to another cell. Table 8.1 lists all the probabilities relating to a single time-step, and these remain invariant throughout all our simulations. These parameter values are those used by Wolff in his work, and have been chosen since they produce the desired Lotka-Volterra predator-prey oscillations for a random

movement model.

Prey reproduction (b_p)	0.40
Satisfied predator reproduction (b_s)	0.40
Hungry predator reproduction (b_h)	0.01
Prey mortality (m_p)	0.05
Satisfied predator mortality (m_s)	0.05
Hungry predator mortality (m_h)	0.08
Prey migration (ν)	0.45
Predator migration (μ)	1.00

Table 8.1: Invariant spatial predator-prey simulation probabilities
Parameter values from Wolff [1989] for a spatial predator-prey system in which predators' reproduction and mortality rates depend upon how recently they ate, i.e. whether they are *hungry* or *satisfied*. This choice of parameters produces Lotka-Volterra oscillations if migration is performed at random to any other location in the system.

In addition to the fixed parameters in Table 8.1, there are two additional system parameters which are varied between simulations as part of the study. The first of these is the *carrying capacity* (K) of a location with respect to the prey species. Lotka-Volterra oscillations are produced without this factor, and cannot be stochastically modelled without an exponential explosion in the populations (see Renshaw [1991]). However, when a carrying capacity was introduced by Volterra [1931] the resulting *Volterra oscillations* do lend themselves to realistic stochastic modelling. The carrying capacity equates to the maximum prey population a location can support in the absence of predators. It therefore represents a limit to natural resources and thus prevents prey populations rising exponentially. To incorporate the carrying capacity, the prey birth routine reduces the probability of reproduction as the prey population rises, so that this probability will reach zero as the population reaches the carrying capacity K, viz:

$$\Pr(\text{birth}) = b_p \times (1 - (\text{Cell Population}/K)) \ . \tag{8.1}$$

Our second variable parameter is termed the *predator-power*, and provides a measure of the average hunting ability of a predator. This can therefore be considered as equivalent to the "voracity" of predators in the work of Bartlett [1957]. The predator-power is implemented as the probability with which a predator will catch a particular prey that is in the same location during a particular time-step. Once a prey is caught, the predator

Parameter	Typical Value	Range Used
Carrying capacity (K)	10	5 – 100
Predator-power (P_p)	0.3	0.2 – 0.4

Table 8.2: Variable spatial predator-prey simulation parameters
Two simulation parameters are varied during our spatial predator-prey work — the prey carrying capacity, and the predator hunting power. This table gives the normal values used, and the range of other values considered during our investigations.

becomes *satisfied* and stops hunting. If the prey escapes, the predator will attempt to catch another prey inhabitant of the field, should one exist. Typical ranges of values for these two parameters are given in Table 8.2.

In order to mimic the non-spatial Volterra model results with our spatial stochastic model, we produce a simulation where a complete mixing of animals takes place at each time-step. In effect, animals are allowed to migrate to any randomly chosen location in the world. Although unrealistic when compared to nature, this simulation is intended to remove the spatial aspects of the system, and thus act as a control to test the suitability of the model for reproducing Volterra's results. The re-shuffle of individuals is implemented such that the field populations lie on a Normal distribution and, using the probabilities given above, it produces the intended oscillations, as detailed later in Section 8.3.

8.2.2 Preferential Migration Simulations

Our basic spatial predator-prey model can be extended to introduce rules for the preferential migration of predators and prey. Thus random infinite migration (i.e. migration to any other location selected at random) is replaced with migration into one of the four neighbouring locations. The choice of cell into which an individual migrates is made either completely randomly, or according to certain preferential migration rules. Predators and prey are therefore divided into two subclasses — "random-movers" and "rule-movers".

In order to investigate the influence of preferential migration on our spatial predator-prey system, the initial populations of predators and prey, in each such simulation, contain just 2-3% of rule-moving animals, the rest being random-movers. There is no method

of inter-breeding between the two subclasses in each species, therefore random-movers always give birth to like animals, as do rule-movers. In addition, preferential migration has no effect on a predator's hunting abilities, or a prey's escaping abilities. Therefore the relative survival of each type of individual is purely dependent upon their migration behaviour. We discuss below the relative success of various preferential migration rules in terms of the growth or decline of the rule-moving proportion of the populations, and the stability of the system as a whole, as the populations evolve. The preferential migration rules investigated are:

Foxdelay The probability of predator migration is weighted by the prey population in the destination location, as a proportion of the total prey population in all four neighbouring locations (see Figure 8.1). Similarly, the rule-moving prey preferentially migrate towards the neighbouring field with the smallest predator population. Foxdelay acquires its name from the asynchronous nature of the population counting and the movement of the species. The migration probabilities use the field populations as they are at the start of the migration routine. However, the prey migrate first, and the predators then migrate with the knowledge of where the prey have already migrated to. Therefore the rule-moving predators have a stronger preferential migration rule than the prey.

Eighthunt Similar to Foxdelay in that with this rule predators again migrate after the prey, and both can migrate to one of the four neighbouring locations. However, now the predators migration probabilities are weighted by studying the prey populations of all eight surrounding fields (see Figure 8.1), and thus encourage movement in the direction of an area that may correspond to that which has a higher potential prey density, i.e. in the direction of the group of three cells with the highest population, rather than a single cell.

Limitfox The Limitfox migration rule is identical to that of Foxdelay, except that the rule-moving predators are restricted from dominating completely by the introduction of a 5% probability that a rule-moving predator will produce offspring that are random-movers.

Synchro As the name suggests, this migration rule has synchronised movement of predators and prey. The Foxdelay probability weighting is used again, but both

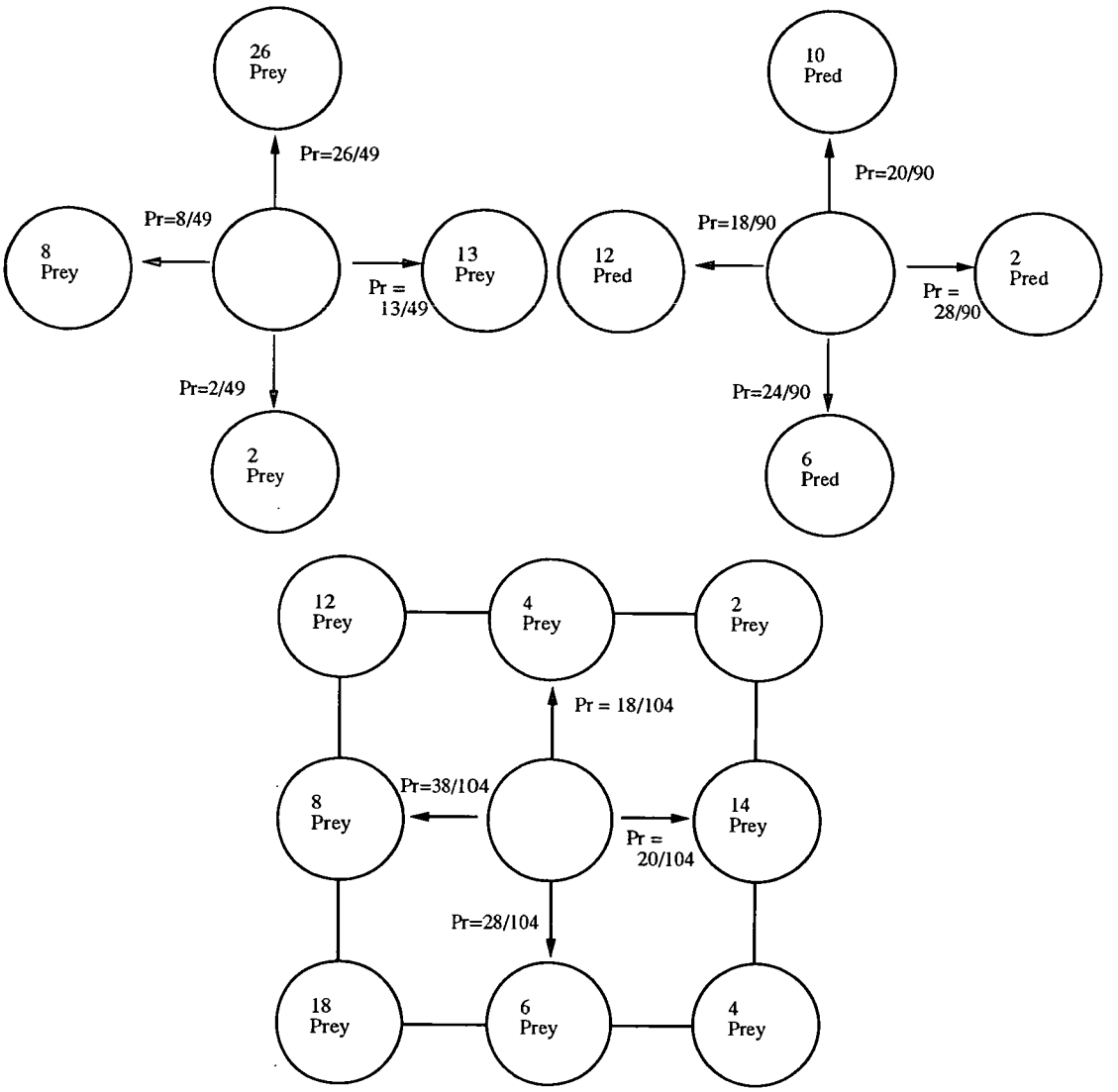

Figure 8.1: Preferential predator migration probabilities

These diagrams show example predator migration weightings for the three rules cases of Foxdelay (top left), Shyfox (top right) and Eighthunt (bottom). The Foxdelay weightings are also used as the basis for the Synchro and Limitfox simulations, and are inverted for the Dumbfox examples.

the predators and prey preferentially migrate using their knowledge of field populations at the end of the previous time-step. The predators therefore have no knowledge of present prey populations, as they do for Foxdelay.

Shyfox This migration rule has a very different basis for the predator migration. The weighting given to the probability of migration into neighbouring locations is now dependent upon the number of predators (rather than prey) in those locations. Rule-moving predators now preferentially move away from other predators, as shown in the example probabilities in Figure 8.1.

Dumbfox In this simulation, rule-moving predators are endowed with a deliberately poor preferential migration rule. The probability weightings of Foxdelay are inverted to encourage rule-movers to migrate into the fields with fewest prey. This deliberately weak migration rule is studied simply as a control on the behaviour of the simulations.

8.2.3 Evolutionary Process Simulation

Since our original model is designed to follow individual members of each species, it is a straightforward task to allow certain attributes of the individuals to vary. We can therefore introduce low-probability small-scale changes to these attributes during the course of a simulation, for example a mutation in a parameter value as it was passed from one generation to the next. Thus over very many generations we may be able to simulate evolutionary forces by studying the distribution of parameters values across all individuals as the simulation develops.

The parameter chosen to be varied in this initial investigation of evolution is the migration probability of the predator species. Therefore predators are divided into ten groups each with a different probability of migration (in equal intervals from 0.1 to 1.0). If this were the only adaption made to the model, it would be expected that the population would simply swing towards the faster (more frequent) movers, since their additional speed will allow them to cover more locations and therefore increase their probability of finding food. We therefore introduce a relationship between the migration probability of a predator and its reproduction capability. As the predator uses more resources to migrate, it has less left to use for reproduction. This produces a balance between the two

"attributes", and most individuals cluster around this "optimum" solution. Ultimately we aim to look at finding the evolutionary stable optimum cluster for a large number of variable parameters.

In this scenario we will have a complex model in which we may introduce a single mutation parameter that describes how far each individual has evolved from its original state. Both species may then alter their migration rate and any other connected parameters (e.g. reproduction rate, predator power, hunting strategy etc.) to any real value according to this overall mutation parameter. Initial implementations of such systems formed the basis of the simulation performance results detailed in the previous chapter. The *complexity* studied in the PPR performance results (see Section 7.3) is directly related to the amount of "evolutionary" calculation to be made for each mutating individual in these simulations. We can therefore begin to study the distributions of individuals that occur within such complex evolving systems, and we are greatly assisted in this work by the dynamic load balancing functions that we have developed.

8.3 Spatial Predator-Prey Simulation Results

The results detailed in this section are divided into three groups: firstly, those that show the production of Volterra-type oscillations within our spatial system; secondly, we give a review of our studies into preferential migration; and thirdly, the results of our evolutionary process simulations.

8.3.1 Volterra Oscillations in Spatial Models

These simulations (as described earlier in Section 8.2) are typically run over 1,500 time-steps using initial populations of 10,000 prey and 6,000 predators, randomly distributed throughout the model world. Figure 8.2 shows the variation with time of the total populations of predators and prey using carrying capacity $K = 10$ and predator power $P_p = 0.4$. The Volterra oscillations produced by this system are quite clear from these graphs, and Figure 8.2 also shows the corresponding limit cycle for these oscillations. We believe that this result confirms that our spatial system simulation can be used as a base for further investigations of more complex ecological systems. The graphs shown

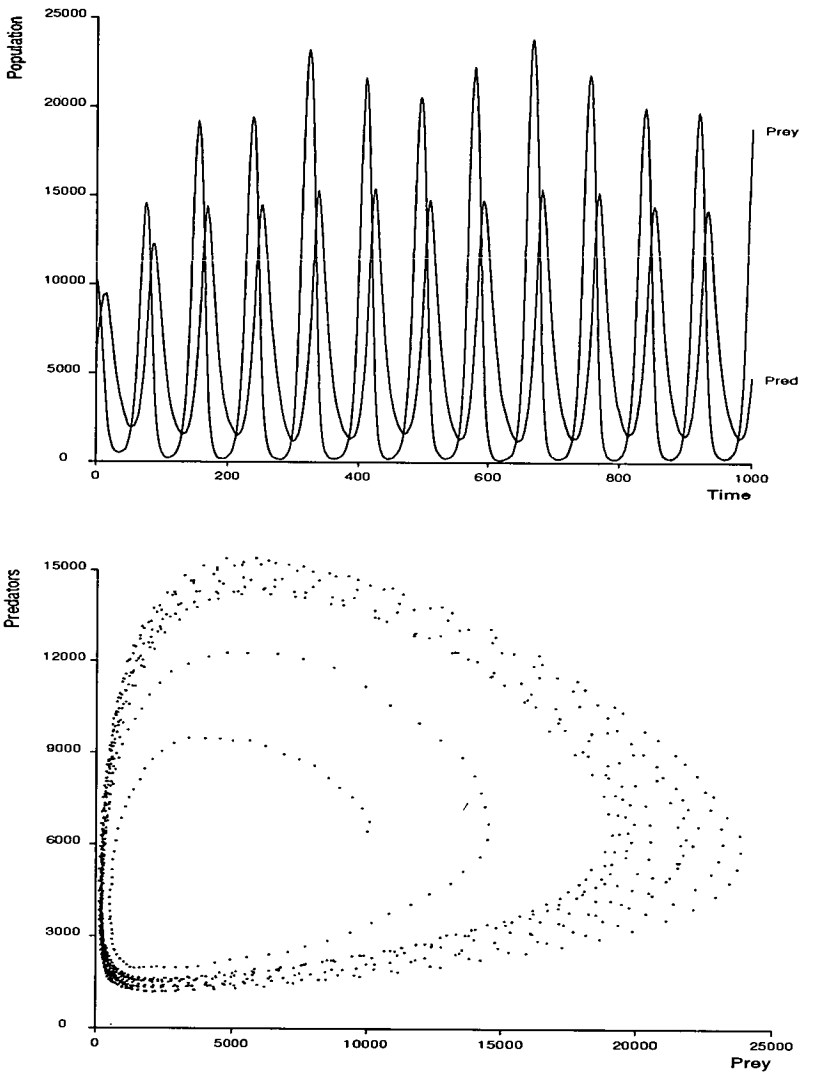

Figure 8.2: Stochastic spatial Volterra population cycles

Variation in total stochastic predator and prey populations with time, using carrying capacity $K = 10$ and predator power $P_p = 0.4$. The top graph shows the total global predator and prey populations, the lower graph shows the corresponding population limit cycle. Note the low variability in the curve shapes between cycles.

in Figure 8.2 show a remarkable regularity in the predator and prey populations. We believe that this is in some part due to the large spatial scale of the model used, allowing global populations to be little effected by localised extremes of behaviour.

Using this same model, but increasing each location's carrying capacity to $K = 15$ has been found to produce more regular and larger amplitude oscillations in both predator and prey populations. In many simulations the prey species is forced into extinction after a number of cycles, as the predator population becomes so large during oscillation peaks that the prey species is eradicated. This result cannot be reproduced by standard deterministic equation models, but requires the demographic stochasticity that this spatial model provides. A similar effect can be produced by holding the carrying capacity at $K = 10$ and raising the predator power to $P_p = 0.5$. This again decreases the chances of prey survival, and therefore when prey are small in number, increases the chance of a distribution of predators occurring that can force the prey into extinction.

An alternative phenomena can be produced by decreasing either K or P_p. Under these conditions a more stable system is produced, and initial population oscillations damp down to give fairly stable populations (see Figure 8.3).

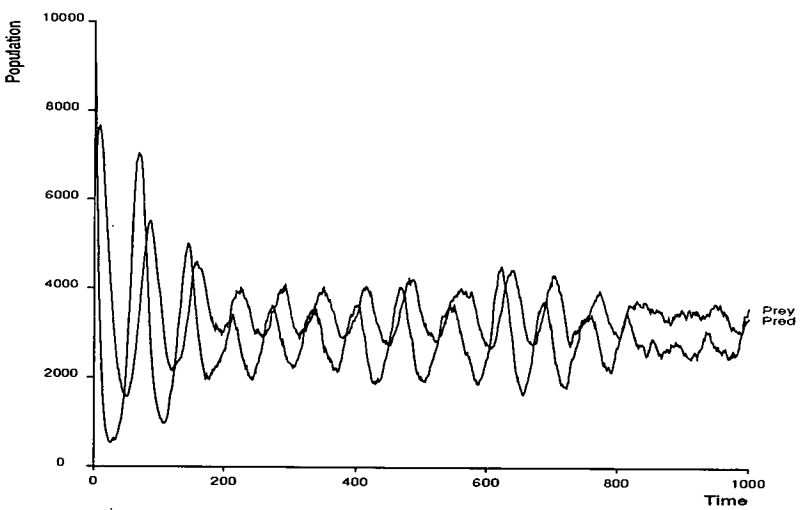

Figure 8.3: Low carrying capacity Volterra oscillations

The Volterra-type predator-prey system is altered by reducing the location carrying capacity to $K = 5$, the predator power is held at $P_p = 0.4$. We observe a distinct damping of population oscillations, due to the reduced maximum number of prey in any single location. This restriction prevents severe population explosions, and thus produces a stable stochastic system that never nears extinction of either species.

8.3.2 Preferential Migration Simulation Results

Following the successful reproduction of typical predator-prey phenomena using our spatial model, we can extend our investigation to study the effects produced by introducing preferential migration rules for both species. Individuals can now move into one of the four neighbouring locations according to the migration rule in use (see Section 8.2). Each of the simulations detailed in this section is run over the same time period (1,500 time-steps), and uses the parameters detailed in Table 8.1. Initially a random distribution of 10,000 random-moving prey and 6,000 random-moving predators is used, with an additional 200 rule-movers from each species scattered randomly throughout the system. For all these runs the variable parameters are set to $K = 10$ and $P_p = 0.3$.

Figure 8.4 shows the simulation results for our standard preferential migration system (Foxdelay) in which members of both species can migrate to the neighbouring location with most prey (for predators) or least predators (for prey). The graphs in Figure 8.4 show the rise to dominance of the rule-moving animals of both species as well as the time evolution of the total populations. It can be seen that the rule-movers of both species rise to dominance, and the eventual extinction of the random-movers. The predators reach extinction much sooner than the prey. This is to some extent due to their advantage in migrating second. It is probably also caused by the predators having a much stronger evolutionary pressure to be "intelligent". This extra "pressure" follows from the fact that by making the correct choice of location a predator has an immediate influence upon its probabilities to survive and reproduce. The prey, on the other hand, have no direct effect on their reproduction, although their survival chances will be reduced as an indirect effect of being better "escapees" from the predators.

The graph of total populations for Foxdelay shows that the model still produces predator-prey type oscillations. It is interesting to note the increase in oscillation amplitude as both species become dominated by rule-movers. It can be seen that the oscillations become most severe when there is the greatest "power" difference between the predator and prey populations, i.e. when the predators have become 100% rule-movers, but with the prey mostly random-movers. Once the prey population nears 100% rule-movers the oscillation amplitude reduces substantially, and thus the global system becomes more stable. This phenomenon occurs with all our migration rules, and has been tested using a variety of initial distributions of predators and prey. None of these influence the result,

since any initial distribution is always destroyed within the first population oscillation.

The Eighthunt migration rule also leads to rule-mover dominance as with Foxdelay, although the rate of increase in the proportion of rule-moving predators is significantly slower (see Figure 8.8 later for a graphical comparison of all the predator migration rules). It would seem that by weighting movement to locations that cannot actually be accessed, the predator migration rule is weakened. This effect makes sense when one considers that predators' overwhelming requirement is for immediate food, in order to gain improved reproduction and survival chances. Thus the strongest preferential migration rule should take predators directly to prey, hence the result that Eighthunt is a weaker predator rule than Foxdelay.

However, for Eighthunt the rule-moving prey rise to dominance much faster than for Foxdelay. This can be explained by considering that in order to undergo a population boom, a prey species must reside in an area relatively devoid of predators. This demographic stochasticity allows localised prey population explosions, as well as the survival of small prey populations. The weakening of the predator hunting rule probably allows the formation of more of these predator-free areas, therefore giving the small number of rule-moving prey the opportunity to dominate more rapidly.

The Limitfox preferential migration rule, where 5% of rule-moving predators revert to random-movers, is another example of a slight weakening of predator power. We find that it also allows the rule-moving prey to dominate slightly faster than for Foxdelay. In addition, the rule-moving predator domination, as well as being limited to around 85% of the total population (the mutation is one-way only), is not so swiftly achieved. It is also interesting to note that the oscillations in total populations are not as severe as for Foxdelay. Figure 8.8 compares the oscillation size of all the migration rules.

In the Synchro preferential migration rule we have now removed all the predator advantage of Foxdelay, as both species migrate based on knowledge of only the population levels at the last iteration. This results in the rise to dominance of the rule-moving prey being greatly accelerated as compared to previous rules, and the rise to dominance of the rule-moving predators taking far longer. The most interesting result of the Synchro simulation can be seen in Figure 8.5. The large amplitude oscillations in total populations disappear almost completely once both predators and prey are all rule-movers. The resulting system is therefore far more stable than for the previous "stronger" pref-

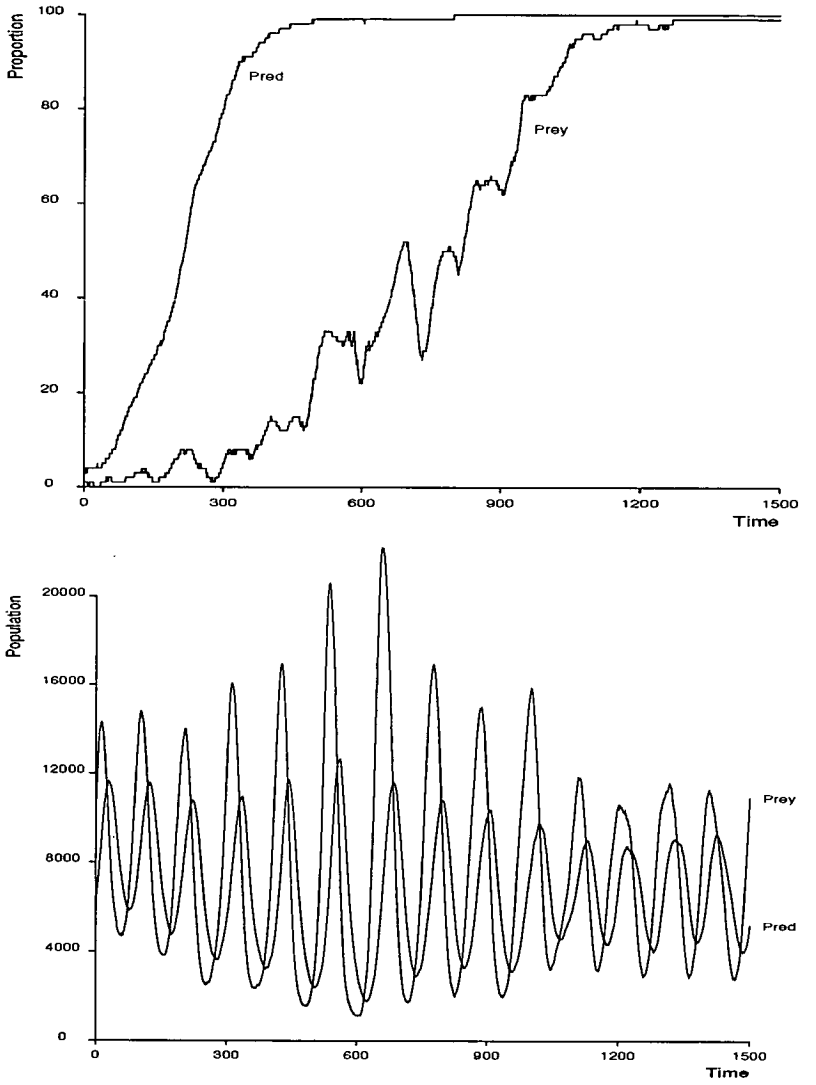

Figure 8.4: Stochastic population development for Foxdelay

Using the Foxdelay migration rule we observe from the top graph that the proportion of rule-movers within each population grows rapidly, with predators reaching dominance within 500 iterations, and prey doing the same by 1,200 iterations. The bottom graph shows the global populations of both predators and prey, and highlights the increase in system instability (oscillation amplitude), as the discrepancy in the strength of each species increases.

erential migration rules, where "stability" is a measure of the movement (away from an equilibrium population) that occurs for each oscillation.

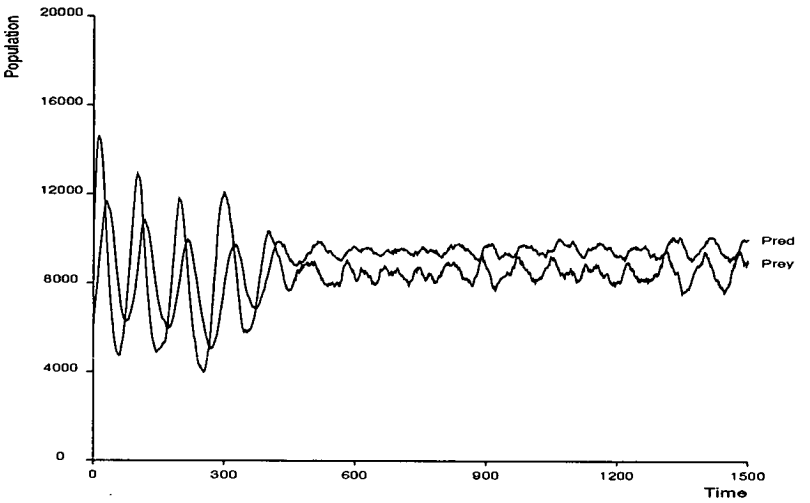

Figure 8.5: Variation in total populations for Synchro

Using the Synchro preferential migration rule, both predators and prey rule-movers base their migration on the positions of each species at the last iteration. This graph shows the variation in total populations within the simulation, and highlights the fact that from iteration 500 onwards the system has become fairly stable. This corresponds to the point when both species have become almost completely Synchro rule-movers.

Our last results within this section are concerned with the Shyfox migration rule. This rule is very different from all others in that rather than predators selecting a migration direction based on the number of available prey, they select on the basis of moving to where there are fewest other predators. This leads to predators spreading out across the world as evenly as possible, thus increasing the hunting chances of the species as a whole, to the possible detriment of the individual. Figure 8.6 shows that preferential migration is still a benefit to both species, as the proportion of rule-movers rises, although the rise to dominance is slowed substantially compared to other rules. It is important to note that the rise in rule-moving Shyfox predators shows that the action of moving to free areas is certainly better than purely random movement. In addition, Figure 8.6 shows that we again have a very stable global system. Once both populations move away from random movement we see the amplitude of global population oscillations being reduced almost to zero. Such a system never moves close to either species becoming extinct.

In this section of results we have detailed a range of stochastic simulations of ecological systems with specific individual behavioural rules. These simulations formed our

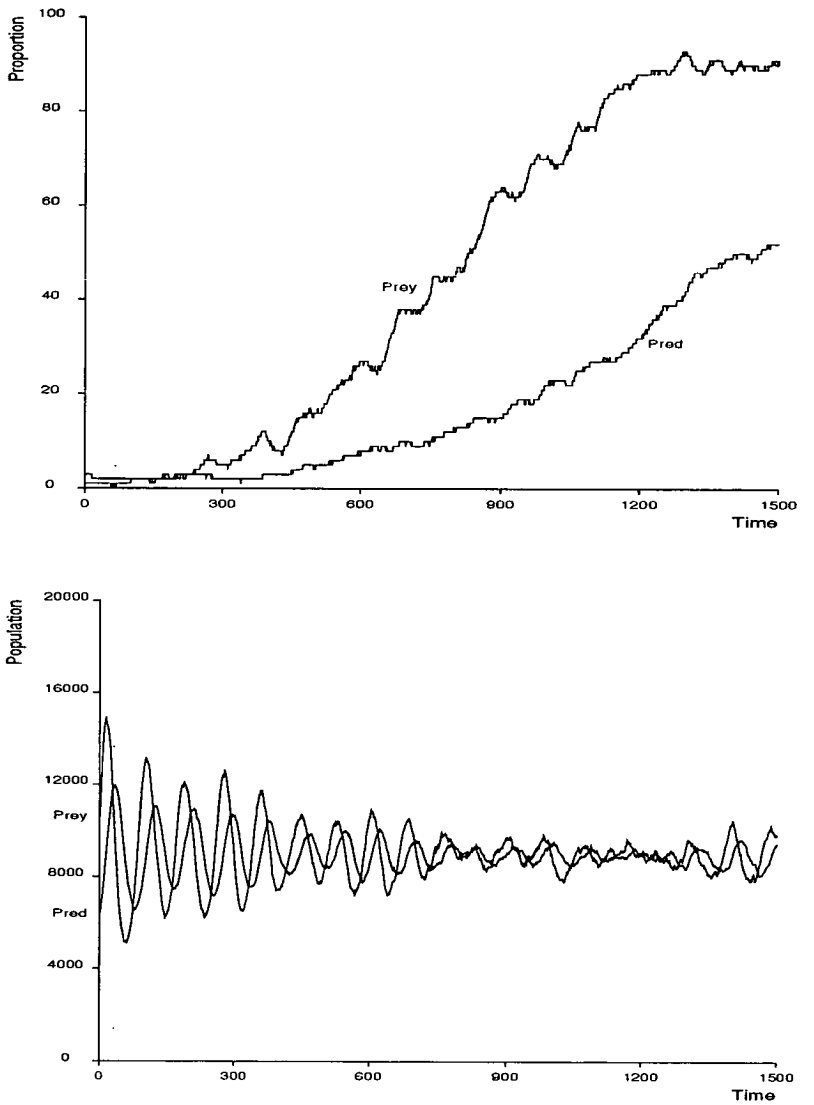

Figure 8.6: Stochastic population development for Shyfox

Using the Shyfox preferential migration rule we see that both predators and prey rule-movers rise slowly in their proportion of the total species populations (top graph). In addition the total species populations (bottom graph) are remarkably stable, showing only very small oscillations.

early studies in this field, and were all performed on a discrete-time basis, using a massively-parallel SIMD supercomputer. None of these simulations use the dynamic load balancing techniques described in the previous chapters, and thus were rather unpredictable and extensive in their use of the supercomputing resources. It was this experience that prompted the development of our advanced modelling techniques in order that our future studies can make more efficient and controlled use of the available machines. The first such "applications" use of these advanced techniques will be in the area of evolutionary modelling, from which we have taken our test-bed performance results.

8.3.3 Evolution Simulation Results

As discussed earlier in this chapter, we first introduce mutation of individual attributes into the spatial predator-prey system running on the DAP-608 parallel computer. In these simulations we allow ten varieties of predator. These varieties differ in terms of the probability that they will migrate in any one time-step, the possible values being equally spaced from 0.1 to 1.0. In order to prevent faster migrating predators from dominating, the predator reproduction rate is adjusted as well as the migration rate. Thus we introduce an adjustment of 0.05 in reproduction probability for every 0.1 change away from the median migration probability, viz:

$$\Pr(\text{birth}) = b_s - 0.05 \times (\mu - 0.5) \ . \tag{8.2}$$

Mutation is included in the simulation by allowing a 2% chance that a reproducing predator will give birth to offspring with a migration probability differing by ± 0.1, and therefore a reproduction probability differing by ± 0.05. One individuals have minimum or maximum migration rates further mutation is only permitted in the one)logical) direction. We find that the mutation function allows dynamic formation of a distribution of populations of the types of predator, a typical example of such a distribution is shown in Figure 8.7.

Using a consistent set of simulation parameters, similar distributions are reached from a wide variety of initial distributions, from the two extremes of almost all fast or all slow predators, and through many intermediate distributions. There is clearly scope

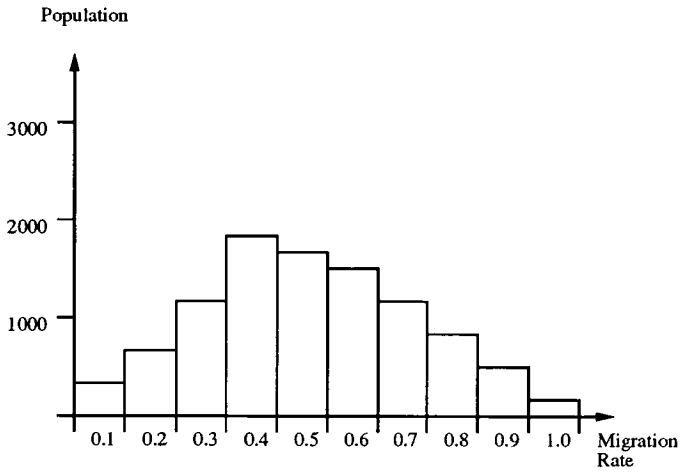

Figure 8.7: Typical population distribution of predator varieties
Our first simulations of evolutionary systems are based on the original Volterra oscillatory system detailed in Section 8.2. However, we now have ten classes of predator, with specific migration and reproduction probabilities. This graph shows the typical steady-state distribution between these classes for a random migration simulation.

to investigate the distributions produced by stochastic models for other many other inter-relations between predator and prey attributes. We now intend to progress our work towards production of a more accurate model of such class distributions. Through use of a flexible simulation system, developed for the Connection Machine CM-200, and using the PPR dynamic load balancing algorithm (see Chapter 7), we intend to develop a much more detailed analysis of an evolutionary spatial predator prey system (as discussed in the previous section) to build upon these initial evolutionary system studies.

8.4 Case Study Conclusions

The results produced by our investigations into various preferential migration rules are summarised in Table 8.3. These cover a single set of parameter values, as no study of the effects produced by varying these parameters was undertaken. Some of the effects observed will be parameter-dependent, rather than model-dependent, however this investigation centres around studying the effects of preferential migration on a model known to exhibit predator-prey oscillations. We feel that these results highlight the important and interesting ecological implications that can be identified through

Name	Pred. Gradient	Oscillation Amp.	Prey Gradient
Foxdelay	1/3	12000	1/12
Limitfox	1/4	10000	1/11
Synchro	1/12	2500	1/4
Eighthunt	1/5	9000	1/8
Shyfox	1/25	2500	1/12

Table 8.3: Summary of preferential migration simulations results
This table details the "strength" of each preferential migration rule, measured as the rate of increase in the proportion of rule-movers in the population, and the "instability" in the resulting system, as measured by the amplitude of the global population oscillations. "Pred. Gradient" is a measure of the rate of rise to dominance of rule-moving predators; the "Prey Gradient" is a similar measure for that species; and "Oscillation Amp" is a measure of the system instability in terms of the average total population oscillations in the later stages of the simulation.

the use of high performance computing. We have developed techniques to allow this exploitation of supercomputers to be more straightforward and efficient. We have not as yet obtained a full set of application results, this will be the subject of future work.

The gradient values given in Table 8.3 are a measure of the strength of the preferential migration rules, in that they are a measure of the speed of the rise to dominance of each of the species. The amplitude of the population oscillations is taken as a measure of the instability of the global system, and is measured from when both populations are dominated by rule-movers.

Figure 8.8 shows the relationship between the strength of the preferential migration rules and the stability of the system. The work of Hilborn [1979] and Zeigler [1977] has shown that both deterministic and stochastic spatial models can bring stable equilibrium to a system. The results given here show that for very large systems with periodic boundary conditions an individual-based stochastic model can run for very long time periods without nearing extinction. The model still exhibits constant population dynamics, both in terms of oscillations in global populations, as well as dynamic population patchiness (see Figure 8.9). In addition, these results show this persistence for a range of predator hunting strategies, and provide some idea of how such strategies affect the stability of the global population.

There appears, even from the few rules investigated, a clear relationship between the strength of the predator hunting rule and the stability of the predator-prey system as

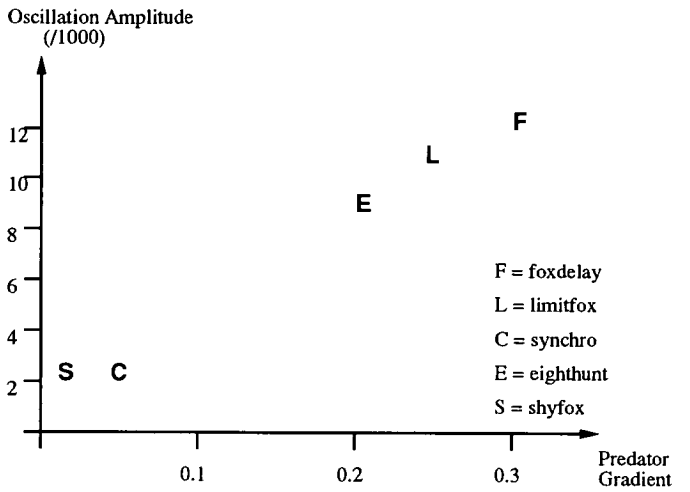

Figure 8.8: Strength versus stability for predator migration rules
This graph shows a plot of predator migration rule strength (equating to dominance rate) against system stability (i.e. amplitude of population oscillations). We see a clear relationship between the two factors, suggesting that strong predators can lead to unstable global systems, and hence increase the potential for extinction.

a whole. The greater the strength of the predators, the more unstable the system. This suggests that the "selfish" optimisation of individual improvement produces instability that could lead to the extinction of both the species in the system. This is obviously a point of substantial interest to the ecological modelling community (see Smith [1991]) and as such provided a substantial driving-force for the development of both the mathematical understanding of such systems, and also the connecting link to their efficient implementation and investigation on the latest range of high performance parallel supercomputers.

Figure 8.10 shows the relationship between the success of the rule-moving predators and prey for each predator migration rule. For both species the success of the rule-movers is measured as the gradient of the increase in the proportion of rule-movers in the population. The prey migration rule is identical for each of the different predator migration rules. It can be seen from the graph that for each of the predator rules where the predators are "chasing" prey, the rule-moving prey dominate the prey population faster as the predator rule weakens. This "weakness" of the predator seems to give the rule-moving prey more chance to gain rapid control and dominate.

The Shyfox migration rule, where predators attempt to spread themselves evenly across the world, does not fit into this scheme. This rule may be "weak" for predators, but

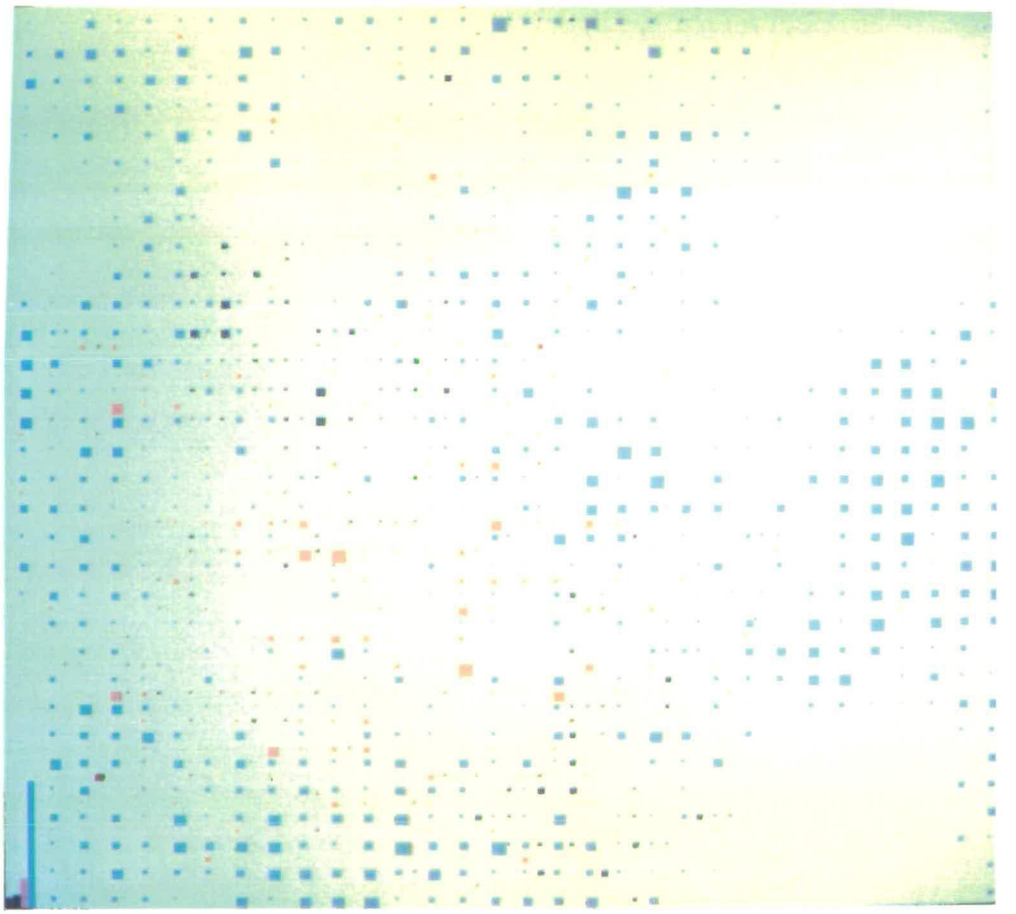

Figure 8.9: Graphical output from spatial predator-prey simulation
Example graphical output from a spatial predator-prey simulation with preferential migration running on an AMT DAP supercomputer. The screen output represents a two-dimensional array of locations. Within each location coloured block represents the population of predators (red blocks) and prey (blue blocks). Light shades of both species represent random-movers, and dark colours represent rule-movers. Note the distinct patchiness in all the populations. This photograph shows the early stages of a simulation, and hence small numbers of rule-moving individuals. The simple bar-graphs in the bottom left corner display the total populations.

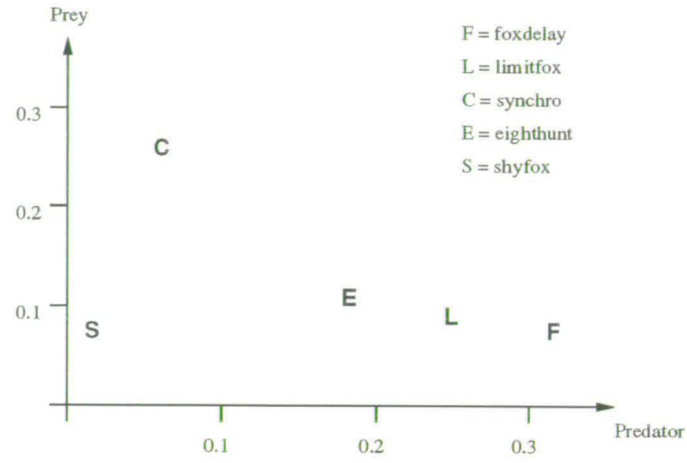

Figure 8.10: Comparison of rates of domination for rule-movers
This graph shows the relative strengths of the preferential migration rules for each predator migration rule. We see a clear inverse relationship between predator strength and prey strength. There is one clear exception to the pattern — Shyfox, in which both predator and prey appear to have "weak" migration rules. This is perhaps best thought of as having a low level of evolutionary pressure to evolve to rule-movement.

it also restricts the rule-moving prey from rapid dominance. Shyfox produces a very stable system, where neither species ever acquires a population dense enough to start a population explosion. This seems to suggest that the Shyfox rule is the best to use should global stability and species survival be the major concerns.

In this work we have only just begun to study the effects of evolutionary processes within ecological simulations. We have however seen that we can produce distribution of individuals within a species, where random mutation has resulted in the species as a whole adapting to fit best its "environment" (i.e. the system parameters and governing equations). There is obviously a substantial amount of potential further work in this area, in particular in expanding the research into "optimising" systems with many more than just two dependent variables.

8.4.1 Ecological Implications

The ability to simulate large spatial systems in short time periods is of great interest to many ecosystem modellers. There is a constant demand for more accurate models of the real world in order to confirm theories on animal behaviour. The patchiness observed in the two-dimensional models described above (see Figure8.9) is similar to

that seen in deterministic models of sea life from the work of Dubois [1975] and Levin & Segel [1976], and the population oscillations produced have been a feature of ecological modelling since the pioneering days of Lotka and Volterra.

One of the important features of this model is that it is stochastic in nature, and exhibits persistent dynamics. It has been shown by Murray [1975] that deterministic realisations of a spatial Volterra system have no permanent patchiness or wave-like properties since any perturbations decay to equilibrium. We know that this is contrary to what is observed in the real world. This fundamental difference between the results produced by deterministic and stochastic models is present through a wide range systems. The deterministic Lotka-Volterra system exhibits damped oscillations, whereas the stochastic version often leads to extinction; the spatial Volterra model gives deterministic decay of perturbations, but stochastic persistence of oscillations (see Renshaw [1991]). While is is somewhat risky to place one's belief completely within a single theory, the merits of stochastic simulations are undeniable.

The case for the use of complex individual-based models has been made by, for example, Conrad & Rizki [1989], and this route is potentially lucrative for ecosystem modellers. As systems become more complex the cost of performing full stochastic realisations can become too expensive in terms of execution time, however the availability of parallel computers can help to make such work possible. This availability will be even more important as our work continues into a deeper study of evolutionary systems. Ever more complex models will be used as mutation of parameters is permitted, and thus more calculations are needed for each individual in the system.

9

Conclusions

Mankind always sets itself only such problems as it can solve; since, looking at the matter more closely, it will always be found that the task itself arises only when the material conditions for its solution already exist, or at least are in the process of formation.

– Karl Marx

9.1 Computational Science and Mathematical Analysis

In this work we have brought together recent advances from computer science and general computational science, and coupled them to new analytical work from mathematics. It is our hope that such efforts will encourage further development within both disciplines. In particular, we feel that as scientific disciplines broaden their range of investigation, they can to look to use research from the other disciplines upon which they begin to impinge. In this work we have started to touch upon many other forms of scientific study. The systems we investigate are of direct relevance to areas of biological, chemical, ecological and physical study. We hope that researchers in these disciplines will also have something to gain from a greater mathematical understanding of the models they develop, and of the computer systems upon which they investigate them.

The spatial reaction systems that form the basis for this work have been studied mostly in the general case. It is intended that this maintains their accessibility to scientists from varied disciplines. In reviewing and expanding upon Turing's [1952] original linearised analysis of such systems, we have introduced the concept of system "kilter" and "criticality". We believe that these categorisations of system behaviour provide an intuitive mechanism to predict the general activity of spatial systems with linear local interaction. This understanding leads to good approximations for non-linear and stochastic activity, and we have stressed, in particular, that this knowledge provides a mechanism for controlling parallel computer implementations of such systems. Our mathematical analysis of two- and three-reactant systems with non-linear interactions focussed specifically on the production of travelling reactant waves. Although our realisation and simulation work showed that such scenarios can be produced by computation, an analysis assuming that such waves move with a constant velocity fails to produce a result. We must assume that the Laplace transform-type power series expansions are not an adequate approximation to transient population waves in spatial reaction systems.

In our mathematical analysis of stochastic spatial reaction systems we developed a mechanism for calculating the expected behaviour of systems with linearised interactions. Through unifying our spatial-temporal covariance analysis over all kilter and criticality regions, we have developed a mechanism for producing exact solutions for the covariance of all perturbations to reactant populations throughout our stochastic

system. Such analysis allows us to predict the expected distribution patterns of reactant "hot-spots", as well as the expected temporal behaviour of such reactant concentrations. All such information proves very useful for the selection of optimal techniques for implementation of simulations of such stochastic systems on parallel computers.

As a major constituent of this work we have developed a suite of computer programs for the realisation and simulation of spatial reaction systems. Many examples of the graphical and statistical output from this software can be viewed in the preceding chapters. In particular in Chapters 3 and 5 we review the behavioural modes of deterministic and stochastic spatial reaction systems. We have concentrated particularly on systems that exhibit permanent wave structures in the reactant populations. This is currently an area of great interest as natural systems that exhibit such effects are being identified in biology (e.g. see Nagorcka & Mooney [1992]) and chemistry (see Ouyang & Swinney [1991]). Our analysis produces a review of the permanent wave scenario, plus new results that link wave amplitude decay rates to system criticality. In addition we provide evidence for the importance of discrete-space reaction systems for the production of certain "real-world" phenomenon. These simulation results help to reinforce the importance of computational techniques as tools for scientific and mathematical investigation.

Our extensive use of high performance computing is inherently linked to our simulation work. In an almost circular relationship, our initial parallel computer investigations produced an interest in spatial reaction systems. This interest led to the use of very large amounts of supercomputer resources for large-scale simulations. The nature of stochastic systems necessitated the development of specialised parallel implementations (incorporating dynamic load balancing techniques) to enable efficient and effective use of such supercomputers. In order to achieve maximum efficiency from these implementations, it was found that we required predictive information about spatial reaction system behaviour. We therefore looked to a specific mathematical analysis of both deterministic and stochastic systems to enable the extraction of such predictive measures. This leads back to being able to use the available supercomputing resources with optimal efficiency, and thereby return to our extensive investigation of the behaviour of complex spatial reaction systems.

9.2 Directions for Future Study

We feel that there are many opportunities for future work in developing a deeper understanding of spatial reaction systems. In particular, further analysis of systems with full non-linear interactions would certainly enhance our current levels of understanding. Analysis of the stochastic system has enabled efficient implementations of spatial reaction system simulations for the two currently dominant types of parallel computer. Hopefully this analysis will also be expanded to gain more theoretical insight into the behaviour of these systems. Our own future study will now, however, concentrate on a study of *evolutionary* spatial reaction systems.

We have already obtained interesting initial results for spatial systems with reactants that "evolve" by adapting their behaviour or physical attributes between generations. One of the basic tenets of Darwinian evolution is that all animals have the capacity to more than replace themselves. Exponential explosions of species populations are prevented in nature by a vast array of factors: predators, limited resources, variable seasons and weather, and also disease. The result of this complex interaction is that almost all populations fluctuate, exhibiting a cyclic pattern of population explosions followed by mass-reductions in numbers with species reaching low densities, often on the point of extinction. It is possible that at these minimum population levels mutation of attributes can have it's strongest effect, as the mass reproduction about to occur may produce proportionally large numbers of mutants in a very short time-scale. It is a major task of modern ecologists to understand the mechanisms involved in these population fluctuations. The interest is not just theoretical, but has practical applications in the prediction of the effects of climate change and man-made alterations to the environment. We hope to continue our investigation into evolutionary ecological systems. In particular we will study the development of individuals in systems where we have a number of behavioural traits in competition, thereby discovering those types of activity that are evolutionary stable. We feel that research of this type may yield very interesting results, and will almost certainly produce suggestions for yet more areas of investigation.

Bibliography

Ahmad, I. & Ghafoor, A, (1991) Semi-distributed load balancing for massively parallel multiprocessor systems. *IEEE Transactions on Software Engineering*, 17(10):987–1004.

Amdahl, G.M, (1967) Validity of the single-processor approach to achieving large scale computing capabilities. In *AFIPS Conference*, volume 30, pages 483–485.

Bailey, N.T.J, (1950) A simple stochastic epidemic. *Biometrika*, 37:193–202.

Bailey, N.T.J, (1963) The simple stochastic epidemic: A complete solution in terms of known functions. *Biometrika*, 50:235–240.

Bailey, N.T.J, (1964) *The Elements of Stochastic Processes*. John Wiley & Sons: New York.

Bailey, N.T.J, (1968) Stochastic birth, death and migration processes for spatially distributed populations. *Biometrika*, 55:189–198.

Barah, A. & Shiloh, A, (1985) A distributed load balancing policy for a multicomputer. *Software: Practice and Experience*, 15(9):901–913.

Bard, J. & Lauder, I, (1974) How well does Turing's theory of morphogenesis work? *Journal of Theoretical Biology*, 45:501–531.

Bartholomew, D.J, (1982) *Stochastic models for social processes*. John Wiley & Sons: Chichester.

Bartlett, M.S, (1955) *An Introduction to Stochastic Processes*. University Press: Cambridge.

Bartlett, M. S, (1957) On theoretical models for competitive and predatory biological systems. *Biometrika*, 44:27–43.

Bartlett, M.S, (1960) *Stochastic Populations Models in Ecology and Epidemiology*. Methuen: London.

Basalla, G, (1988) *The Evolution of Technology*. University Press: Cambridge.

Boglaev, Y.P, (1992) Exact dynamic load balancing of MIMD architectures with linear programming algorithms. *Parallel Computing*, 18:615–623.

Boillat, J.E, (1990) Load balancing and Poisson equation in a graph. *Concurrency: Practice and Experience*, 2(4):289–313.

Bryant, R.M. & Finkel, R.A, (1981) A stable distributed scheduling algorithm. In *Proceedings of Second International Conference on Distributed Computing Systems*, pages 314–323.

Bunow, B, Kernevez, J.P, Joly, G, & Thomas, D, (1980) Pattern formation by reaction-diffusion instabilities: Application to morphogenesis in *drosophila*. *Journal of Theoretical Biology*, 84:629–649.

Castets, V, Dulos, E, Boissonade, J, De Kepper, P, (1990) Experimental evidence of a sustained standing Turing-type nonequilibrium chemical pattern. *Physical Review Letters*, 64(24):2953–2956.

Chantler, L.K. (Editor), (1993) *Edinburgh Parallel Computing Centre Annual Report 1992-93*. The University of Edinburgh.

Chow, Y. & Kohler, W.H, (1979) Models for dynamic load balancing in a heterogeneous multiple processor system. *IEEE Transactions on Computers*, 25(5):354–361.

Cliff, A.D. & Ord, J.K, (1975) Model building and the analysis of spatial pattern in human geography. *Proceedings of the Royal Statistical Society*, B 37:297–348.

Conrad, M. & Rizki, M, (1989) The artificial world's approach to emergent evolution. *Biosystems*, 23(2(3)):247–260.

Cox, D.R. & Miller, H.D, (1965) *The Theory of Stochastic Processes*. Methuen: London.

Daniels, H.E, (1954) Saddlepoint approximations in statistics. *The Annals of Mathematical Statistics*, 25:631–650.

Daniels, H.E, (1975) The deterministic spread of a simple epidemic. In *Perspectives in Probability and Statistics*, Ed. J Gani, pages 373–386. Academic Press: London.

Daniels, H.E, (1977) The advancing wave in a spatial birth process. *Journal of Applied Probability*, 14(4):689–701.

Darwin, C.R, (1859) *The Origin of Species*. John Murray: London.

Dawkins, R, (1976) *The Selfish Gene*. University Press: Oxford.

Dawkins, R, (1986) *The Blind Watchmaker.* Longman: London.

den Boer, P.J, (1981) On the survival of populations in a heterogeneous and variable environment. *Oecologia*, 50:39–53.

Denton, M, (1985) *Evolution: A Theory in Crisis.* Burnett Books: London.

Dijkstra, E.W, (1965) Solution to a Problem in Concurrent Programming Control. *Communications of the ACM*, Volume 8.

Dubois, D.M, (1975) A model of patchiness for prey-predator plankton populations. *Ecological Modelling*, 1:67–80.

Eager, D.L, Lazowska, E.D, & Zahorjan, J, (1986a) Adaptive load sharing in homogeneous distributed systems. *IEEE Transactions on Software Engineering*, 12(5):662–674.

Eager, D.L, Lazowska, E.D, & Zahorjan, J, (1986b) A comparison of receiver-initiated and sender-initiated adaptive load sharing. *Performance Evaluation*, 6:53–68.

Edelman, G.M, (1989) *Neural Darwinism: the theory of neuronal group selection.* University Press: Oxford.

Einstein A, (1905) Über die molekular-kinetischen Theorie.... *The Annals of Physics (Leipzig)*, Volume 17.

Ellis, T.M.R, (1990) *Fortran 77 Programming: With an Introduction to the Fortran 90 Standard.* Addison-Wesley: England.

Feller, W, (1939) Die Grundlagen der Volterraschen Theorie des Kampfes ums Dasein in der wahrscheinlichkeitstheoretischen Behandlung. *Acta Biotheoretica*, 5:1–40.

Fisher, R.A, (1930) *The Genetic Theory of Natural Selection.* Clarendon Press: Oxford.

Fisher R.A, (1937) The wave of advance of advantageous genes. *Annals of Eugenics*, 7:355–369.

Flynn, M.J, (1972) Some computer organisations and their effectiveness. *IEEE Transactions on Computers*, C(21):948–960.

Foster, D, (1989) *Modelling evolutionary processes on a computing surface.* Technical Note 27, Edinburgh Parallel Computing Centre, The University of Edinburgh.

Fox, G. & Otto, S, (1986) Concurrent computation and the theory of complex systems. In *Proceedings of SIAM Hypercube Multiprocessors Conference* 1986, p. 244.

Fox, G.C, Johnson, M, Lyzenga, G.A, Otto, S.W, Salmon, J.K, & Walker, D.W, (1988) *Solving Problems on Concurrent Processors (Volume 1).* Prentice-Hall International: London.

Gard, T.C, (1988) *Introduction to Stochastic Differential Equations.* Marcel Dekker: New York.

Gardiner, C.W, (1985) *Handbook of Stochastic Methods.* Springer-Verlag: Berlin-Heidelberg.

Gause, G.F, (1935) Experimental demonstration of Volterra's periodic oscillations in the numbers of animals. *Journal of Experimental Biology*, 12:44–48.

Gear, C.W, (1971) *Numerical Initial Value Problems in Ordinary Differential Equations.* Prentice-Hall: New Jersey.

Greenspan, D, (1960) *Theory and solution of ordinary differential equations.* Macmillan: New York.

Haefner, J.W, (1991) Parallel computers and individual-based models: An overview. In DeAngelis & Gross, editor, *Organisms, populations, and communities: A perspective from individual-based models.* Springer-Verlag: New York.

Hong, J, Tan, X, & Chen, M, (1988) From local to global: An analysis of nearest neighbour balancing on a hypercube. *Performance Evaluation Review, ACM SIGMETRICS*, 16:73–82.

Hayes, J.P, Mudge, T, Stout, Q.F, Colley, S, & Palmer, J, (1986) A Microprocessor-based Hypercube Supercomputer. *IEEE Micro*, 6(5):6–17.

Hengeveld, R, (1989) *Dynamics of Biological Populations.* Chapman and Hall: London.

Hilborn, R, (1979) Some long-term dynamics of predator-prey models with diffusion. *Ecological Modelling*, 6:23–30.

Hodges, A, (1983) *Alan Turing: the enigma.* Burnett Books, Hutchinson Publishing Group: London.

Holland, J.H, (1975) *Adaption in Natural and Artificial Systems.* University of Michigan Press: Ann Arbor.

Hoyle, F, (1987) *Mathematics of Evolution.* University College Press: Cardiff.

Huffaker, C.B, (1958) Experimental studies on predation: dispersion factors and predator-prey interactions. *Hilgardia*, 27:343–383.

Hunding, A, (1990) *Drosophila* segmentation: Supercomputer simulation of prepattern hierarchy. *Journal of Theoretical Biology*, 145:369–384.

Hwang, K, Croft, W.J, Goble, G.H, Wah, B.W, Briggs, F.A, Simmons, W.R, & Coates, C.L, (1982) A Unix-based local computer network with load balancing. *Computer*, 15(4):55–64.

Isham, V, (1988) Mathematical modelling of the transmission dynamics of HIV infection and AIDS: a review. *Proceedings of the Royal Statistical Society*, A 151(1):5–30.

Ito, K, (1951) On stochastic differential equations. In *American Mathematical Society Memoirs No. 4*, New York.

Kauffman, S.A, Shymko, R, & Trabert, K, (1978) Control of sequential compartment formation in *drosophila*. *Science*, 199:259–270.

Kaufmann, W.J. & Smarr, L.L, (1993) *Supercomputing and the Transformation of Science*. Scientific American Library: New York.

Keane, J.A, (1993) Parallelising a financial system. *Future Generation Computer Systems*, 9:41-51.

Kendall, D.G, (1949) Stochastic processes and population growth. *Journal of the Royal Statistical Society*, B 11:230–264.

Koelbel, C.H, (1994) *The High Performance Fortran Handbook*. MIT Press: Cambridge, Massachusetts.

Levin, S.A. & Paine, R.T, (1974) Disturbance, patch formation, and community structure. *Proceedings of the National Academy of Sciences, USA*, 71:2744–2747.

Levin, S.A. & Segel, L.A, (1976) Hypothesis for origin of plankton patchiness. *Nature*, 259:659.

Lin, F.C.H. & Keller, R.M, (1984) Simulated performance of a reduction-based multiprocessor. *Computer*, 17(7):70–81.

Lin, F.C.H. & Keller, R.M, (1987) The gradient model load balancing method. *IEEE Transactions on Software Engineering*, 13(1):32–38.

Lotka, A.J, (1925) *Elements of Mathematical Biology*. Dover: New York.

McKean H.P, (1975) Application of Brownian Motion to the equation of Kolmogorov-Petrovskii-Piscunov. *Communications on Pure and Applied Mathematics*, 28:323–331.

Maini, P.K, Benson, D.L. & Sherratt, J.A, (1992) Pattern formation in reaction-diffusion models with spatially inhomogeneous diffusion coefficients. *IMA Journal of Mathematics Applied in Medicine & Biology*, 9(3):197–213.

May, R.M, (1976) Simple mathematical models with very complicated dynamics. *Nature*, 261:459–467.

May, R.M, (1986) When two and two don't make four: non-linear phenomena in ecology. *Proceedings of the Royal Society*, B 228:241.

Maynard-Smith, J. & Sondhi, K.C, (1961) The arrangement of bristles in *drosophila*. *Journal of Embryology and Experimental Morphology*, 9:661–672.

Maynard-Smith, J, (1982) *Evolution and the Theory of Games*. University Press: Cambridge.

Meiko Ltd., (1990) *CSTools Documentation*, version 1.09 edition, Bristol, UK.

Metropolis, N, Rosenbluth, A, Rosenbluth, M. Teller, A, & Teller, E, (1953) Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6):1087–1092.

Mollison, D, (1972a) Possible velocities for a simple epidemic. *Advances in Applied Probability*, 4:233–257.

Mollison, D, (1972b) The rate of spatial propagation of simple epidemics. In *Proceedings of the 6th Berkeley Symposium on Mathematics, Statistics and Probability*, volume 3, pages 579–614.

Mollison, D, (1977) Spatial contact models for ecological and epidemic spread. *Proceedings of the Royal Statistical Society*, B 39(3):283–326.

Mollison, D, Isham, V. & Grenfell, B, (1994) Epidemics: Models and Data. *Journal of the Royal Statistical Society*, A 157(1):115–149.

Morgan, B.J.T, (1984) *Elements of Simulation*. Chapman and Hall: London.

Murray, J.D, (1975) Non-existence of wave solutions for the class of reaction-diffusion equations given by the Volterra interacting-population equations with diffusion. *Journal of Theoretical Biology*, 52:459–469.

Murray, J.D, (1981) A pre-pattern formation mechanism for animal coat markings. *Journal of Theoretical Biology*, 88:161–199.

Murray, J.D, (1982) Parameter space for Turing instability in reaction diffusion mechanisms: A comparison of models. *Journal of Theoretical Biology*, 98:143–163.

Murray, J.D, (1989) *Mathematical Biology*. Springer-Verlag: Berlin.

Nagorcka, B.N. & Mooney, J.R, (1985) The role of a reaction-diffusion system in the initiation of primary hair follicles. *Journal of Theoretical Biology*, 114:243–272.

Nagorcka, B.N. & Mooney, J.R, (1992) From stripes to spots: Prepatterns which can be produced in the skin by a reaction-diffusion system. *IMA Journal of Mathematics Applied in Medicine & Biology*, 9(4):249–267.

Ni, L.M, Xu, C. & Gendreau, T.B, (1985) A distributed drafting algorithm for load balancing. *IEEE Transactions on Software Engineering*, 11(10):1153–1161.

Nicol, D.M, (1992) Communication efficient global load balancing. *Proceedings of the 1992 Scalable High Performance Computing Conference*, pages 292–299.

Nicol, D.M. & Reynolds, P.F, (1990) Optimal dynamic remapping of data parallel computations. *IEEE Transactions on Computers*, 39(2):206–219.

Nicol, D.M. & Saltz, J.H, (1988) Dynamic remapping of parallel computations with varying resource demands. *IEEE Transactions on Computers*, 37(9):1073–1087.

Nicol, D.M. & Saltz, J.H, (1990) An analysis of scatter decomposition. *IEEE Transactions on Computers*, 39(11):1337–1345.

Nicol, D.M, Saltz, J.H, & Townsend, J, (1989) Delay point schedules for irregular parallel computations. *International Journal of Parallel Programming*, 18(1):69–90.

Onstad, D.W, (1988) Population-dynamics theory: The roles of analytical simulation and supercomputer models. *Ecological Modelling*, 43:111–124.

Ortega, J.M, (1970) *Iterative solution of nonlinear equations in several variables*. Academic Press: New York.

Ouyang, Q. & Swinney, H.L, (1991) Transition from a uniform state to hexagonal and striped Turing patterns. *Nature*, 352:610–612.

Patterson, D.A. & Séquin, C.H, (1982) A VLSI RISC. *Computer*, Volume 15(9).

Press, W.H, Editor, (1988) *Numerical Recipes in C: The art of scientific computing*. University Press: Cambridge.

Renshaw, E, (1972) Birth, death and migration processes. *Biometrika*, 59:49–59.

Renshaw, E, (1973) Interconnected population processes. *Journal of Applied Probability*, 10:1–14.

Renshaw, E, (1977) Velocities of propagation for stepping-stone models of population growth. *Journal of Applied Probability*, 14:591–597.

Renshaw, E, (1981) Waveforms and velocities for models of spatial infection. *Journal of Applied Probability*, 18:715–720.

Renshaw, E, (1982) The development of a spatial predator-prey process on interconnected sites. *Journal of Theoretical Biology*, 94:355–365.

Renshaw, E, (1984) Competition experiments for light in a plant monoculture: An analysis based on two-dimensional spectra. *Biometrics*, 40:717–728.

Renshaw, E, (1986) A survey of stepping-stone models in population dynamics. *Advances in Applied Probability*, 18:581–627.

Renshaw, E, (1991) *Modelling Biological Populations in Space and Time.* University Press: Cambridge.

Renshaw, E, (1994) The linear spatial-temporal interaction process and its relation to $1/\omega$-noise. *Journal of the Royal Statistical Society*, B 56(1):75–91.

Renshaw, E, (1994) Non-linear waves on the Turing ring. *The Mathematical Scientist*, 19:27-51.

Renshaw, E. & Smith, M, (1991) Efficient implementation of stochastic Turing ring systems on parallel computers. In *Proceedings of the 19th European Meeting of Statisticians*, Barcelona, Spain.

Rice, J.R, (1983) *Numerical methods, software, and analysis.* McGraw-Hill: London.

Richardson, L.F, (1922) *Weather Prediction by Numerical Process.* University Press: Cambridge (Republished by Dover Publications: New York (1965)).

Root-Bernstein, R.S, (1989) *Discovering.* University Press: Harvard.

Sedgewick, R, (1988) *Algorithms.* Addison-Wesley Publishing: London.

Simberloff, D.S, (1976) Experimental zoogeography of islands: effects of island size. *Ecology*, 57:629–648.

Skellam, J.G, (1951) Random dispersal in theoretical populations. *Biometrika*, 38:196–218.

Shiyom, M, (1980) Predation-affected spatial pattern changes in a prey population. *Ecological Modelling*, 11:1–14.

Smith, M, (1991) *The Transputer and Its Offspring*, In Past, Present, Parallel: A survey of available parallel computing systems, Trew & Wilson (Eds.), Springer-Verlag: London.

Smith, M, (1991) Using massively-parallel supercomputers to model stochastic spatial predator-prey systems. *Ecological Modelling*, 58:347–367.

Smith, M, (1993) Dynamic load-balancing strategies for data parallel implementations of reaction-evolution-migration systems. *International Journal of Modern Physics C*, 4(1):107–119.

Smith, M, Carmichael, N, Reid, I, & Bruce, C, (1991) Lithofacies determination from wire-line log-data using a distributed neural network. In *IEEE 1991 Workshop on Neural Networks for Signal Processing*, pages 483–492.

Smith, M. & Renshaw, E, (1993) Parallel-prefix remapping for efficient data-parallel implementation of unbalanced simulations. *Advances in Parallel Computing*, in press.

Smith, M. & Wilson, G.V, (1991) *Dynamic load-balancing on a one-dimensional mesh.* Technical Note EPCC-TN91-11, Edinburgh Parallel Computing Centre, The University of Edinburgh. Presented at the Second European Distributed Memory Computing Conference, Munich, Germany.

Smith, R.H. & Mead, R, (1980) The dynamics of discrete-time stochastic models of population growth. *Journal of Theoretical Biology*, 86:607–627.

Sneddon, N.I, editor, (1976) *Encyclopedic Dictionary of Mathematics for Engineers and Applied Scientists.* Pergamon Press: Oxford.

Stone, H.S, (1977) Multiprocessor scheduling with the aid of network flow algorithms. *IEEE Transactions on Software Engineering*, 3(1):85–93.

Stratonovich R.L, (1966) A new representation for stochastic integrals and equations. *SIAM Journal on Control*, 4:362–371.

Traynor, C.A., Anderson, J.B. & Boghosian, B.M, (1991) A quantum Monte Carlo calculation of the ground state of the hydrogen molecule. *Journal of Chemical Physics*, 95(10):7418–7425.

Trew, A.S. & Wilson, G.V. (Eds.), (1991) *Past, Present, Parallel: A survey of available parallel computing systems.* Springer-Verlag: London.

Trewin, S.M, (1992) *PUL-SM prototype functional specification.* Technical Report EPCC-KTP-PUL-SM-PROT-FS, Edinburgh Parallel Computing Centre, The University of Edinburgh.

Turing, A.M, (1936) On computable numbers with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 2(42):230–265.

Turelli, M, (1977) Random environments and stochastic calculus. *Theoretical Population Biology*, 12:140–178.

Turelli, M, (1978) A reexamination of stability in randomly varying versus deterministic environments with comments on the stochastic theory of limiting similarity. *Theoretical Population Biology*, 13:244–267.

Turing, A.M, (1952) The chemical basis of morphogenesis. *Philosophical Transactions of the Royal Society of London*, B 237:37–72.

UKQCD Collaboration, (1992) First results from the UKQCD collaboration. *Journal of Nuclear Physics*, B(26):211–216.

Vibert, J, (1994) Inter-Neural delay modification synchronises biologically plausible neural networks. *Neural Networks*, in press.

Volterra, V, (1926) Fluctuations in the abundance of a species considered mathematically. *Nature*, 118:558–560.

Volterra, V, (1931) *Lecons sur la Théorie Mathématique de la Lutte pour la Vie.* Gauthier-Villars: Paris.

von Neumann, J, (1963) *Design of Computers, Theory of Automata and Numerical Analysis*, In Collected Works (ed. A.H. Taub), Permagon Press: New York.

Wallace, D.J, (1988) Scientific computation on SIMD and MIMD machines. *Philosophical Transactions of the Royal Society of London*, A 326:481–498.

Wang, Y. & Morris, R, (1985) Load sharing in distributed systems. *IEEE Transactions on Software Engineering*, 34(3):204–217.

Wheeler, M.F. (Ed.), (1988) *Numerical Simulation in Oil Recovery.* Springer: New York.

Williams, R.D, (1991) Performance of dynamic load balancing algorithms for unstructured mesh calculations. *Concurrency: Practice and Experience*, 3(5):457–481.

Wolff, W, (1989) Population dynamics. In *Computersimulation in der Physik, 20^{th} Ferienkurs auf Institut fur Festkorperforschung*, Julich, Germany.

Wolpert, L, (1969) Positional information and the spatial pattern of cellular differentiation. *Journal of Theoretical Biology*, 25:1–47.

Zeigler, B.P, (1977) Persistence and patchiness of predator-prey systems induced by discrete event population exchange mechanisms. *Journal of Theoretical Biology*, 67:637–713.