Error Thresholds and Optimal Mutation Rates in Genetic Algorithms

Gabriela Ochoa

Submitted for the degree of DPhil

University of Sussex

December, 2000

Declaration

I hereby declare that this thesis has not been submitted, either in the same or different form, to this or any other university for a degree.

Signature:

Acknowledgements

I am very grateful to my supervisors Hilary Buxton and Inman Harvey for their constant guidance and support. They followed the whole process very closely and suggested useful ideas and insights all the way long. The frequent and stimulating discussions we had at our regular meetings made it all happen. Many thanks to both of you.

Many thanks to my examiners Adrian Thompson and Terry Fogarty for their very useful comments and critical reading.

I am also grateful to my husband, Andy, who not only gave me constant emotional support and love, but also encouraged me to pursue this goal in the first place. He was patient and understanding enough to overcome the difficult period of separation in the first stage of my DPhil. For him the deepest thoughts and love.

I specially want to thank Margarita Sordo for her constant friendship, encouragement and help throughout the whole DPhil. She was really generous in her attitude to help in many ways, and also carefully and critically read a good part of this document.

I am grateful for funding from CONICIT and Universidad Simon Bolivar, Caracas, Venezuela.

Finally, many thanks to my parents Marta and Hernan, my brothers, my beloved grandmother Teodora and my aunt Gisela; for their unconditional love and support, and for being so kind whenever I was back home.

Error Thresholds and Optimal Mutation Rates in Genetic Algorithms

Gabriela Ochoa

Summary

When applying a genetic algorithm to solve a given problem, the designer faces a large number of choices, with little theoretical guidance and few rules of thumb about how to proceed. Among these choices, the setting of evolutionary parameters (e.g. mutation rate, recombination rate, population size and selection parameters) is important since their values determine the performance of the algorithm to a great extent. However, finding a good combination of parameters is not an easy task since they interact with one another non-linearly and cannot be optimised one at a time. Moreover, 'optimal' parameter settings are believed to be problem-dependent. The mutation rate is acknowledged as one of the most sensitive parameters, so good heuristics for setting the mutation rate are welcomed.

This thesis brings the fundamental notion of the *error thresholds* of replication from molecular evolution into the field of evolutionary computation. Error thresholds are intuitively related to the idea of an optimal balance between exploration and exploitation in genetic search. So, it is hypothesised and empirically demonstrated here, that error thresholds are related to the more familiar notion of optimal mutation rates in GAs. This finding sheds new light on the sensitivity of the mutation rate and points toward useful heuristics for setting this parameter. Some results on the effects and usefulness of recombination are also presented. This dissertation also introduces *consensus sequence* plots, which are adapted from theoretical biology, as a new visualisation tool to the genetic algorithms community. They are used for locating error thresholds on general landscapes, and are shown to reveal several features of the landscape structure. The insights and empirical evidence gathered here support a heuristic that sets a rate based on one mutation per genotype, to be scaled according to the selection pressure and also potentially modified for very redundant genotypes. However, since the selection pressure can be controlled, this rule is shown to hold over a wide range of problem types.

Submitted for the degree of DPhil

University of Sussex

December, 2000

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Darwin's theory of evolution through natural selection is a marvellous scientific idea; simple, yet powerful, it is able to explain the origins and diversity of life on Earth. It also explains the broad range of complex adaptations of living organisms to their environment. The process of natural evolution only requires four basic conditions: A population or group of entities, variation among the members of the population, hereditary transmission between parents and offspring, and a sorting process that changes the proportion of different individuals within the population over the generations. Chief among these sorting processes are *chance* (random variation in the survival or reproduction of different variants) and *natural selection* (consistent, non-random differences among variants in their rates of survival or reproduction) (Futuyama, 1998). All these components, considered at an abstract level, may be easily implemented in a computer program giving rise to artificial evolutionary systems. In the 1950s and 1960s several computer scientists independently studied evolutionary systems with the idea that evolution could be used as an optimisation tool for engineering problems. Among these early approaches, three major methodologies have consolidated over the last three decades: *Evolutionary Programming* (Fogel et al., 1966), *Evolution Strategies* (Rechenberg, 1973), and *Genetic Algorithms* (Holland, 1975).

Genetic Algorithms (GAs) are stochastic search methods that mimic the process of natural evolution. They maintain a population of potential solutions to a given problem (*individuals* or *genotypes*). The ability of each individual to solve the problem is measured by a *fitness function*. To simulate evolution, the population is subject to genetic variation (*mutation* and *recombination*) and survival of the fittest (*selection*) through an iterative process that generates increasingly better solutions. Besides classic applications in function and combinatorial optimisation, GAs have been applied successfully in a wide range of real-world domains including the design of telecommunication networks, computer programs, electronic circuits, robot controllers, and biochemical drugs. Applications oriented research is quite successful and dominates the field (if one considers the number of published papers). In contrast, the theoretical foundations are still weak. Consequently, new users fall repeatedly in the same traps, because there are only few rules of thumb for GA design and parameterisation.

Research attempting to improve the design and parameterisation of GAs can be focused along two lines: theoretical and empirical (Fogel, 1995). The theoretical approach seeks to discover mathematical truths about algorithms which will, hopefully, hold over a broad application domain. The empirical approach attempts to assess algorithm performance in specific domains through statistical means. Both procedures are inherently limited. Mathematical proofs about the properties of algorithms may seem at first sight more powerful than mere empirical evidence. EAs, however, incorporate complex non-linear stochastic process. To make formal analysis tractable, the actual algorithm has to be simplified. Moreover, only very simple toy-problems are tackled, missing the complexities of real-world domains. Hence, one may legitimately question the practical relevance of such theoretical studies. On the other hand, through an empirical approach, algorithms can be tested over successive trials on a specific problem, and a statistical estimation of their performance can be determined. Although the performance of an algorithm on one sample problem may not convey general information, the position adopted in this thesis is that it is possible to induce general properties of complex algorithms by assessing their performance across a variety of landscape structures. Moreover, with this approach, and assuming that landscapes can be characterised by certain features, specific algorithm properties may be identified for different kinds of landscapes.

Another source of inspiration towards understanding and improving the practice of GAs is still natural evolution, and particularly molecular evolution. Molecular evolution has been a source of inspiration for evolutionary computation techniques. As a concrete example, molecular biologists have discovered non-functional sequences of DNA, called *introns*; the work of Levenick (1991, 1999) and others (Wu & Lindsay, 1995), demonstrates that the insertion of introns into bit strings can improve the performance of GAs. In a similar vein, this thesis attempts to bring the notion of *error threshold*[1] from the field of molecular evolution to the field of GAs. It also seeks to assess the relevance and potential practical applications of this notion in the context of GAs. Specifically, research from molecular evolution suggests that:

> The speed of the [evolutionary] optimization can be tuned by the replication precision. Optimization will be fastest close to the error threshold, since too exact copying reduces the chance of producing new advantageous mutants. In nature, a number of viruses have been shown to operate close to their error threshold. This enhances their flexibility to adapt to a continuously changing fitness landscape. [(Bonhoeffer & Stadler, 1993), p. 365].

Moreover, analytical expressions of the error threshold on simple landscapes, suggest how other components in the evolutionary process will affect this threshold. Hence, this thesis postulates that this knowledge may provide useful insights into the design of effective GAs, and may help to predict the effects and interactions of evolutionary parameters in the search process.

1.1 Quasispecies and Error Thresholds

Quasispecies theory was derived by Eigen and Schuster (1979) to describe the dynamics of replicating nucleic acid molecules under the influence of mutation and selection. The theory was originally developed in the context of pre-biotic evolution (studies of the origin of life), but in a wider sense it describes any population of reproducing organisms. A quasispecies is defined as the stationary population distribution of replicating macromolecules under mutation and selection.

[1]The error threshold of replication is defined as the minimal replication accuracy necessary to maintain the genetic information in the population.

The most prominent feature of the quasispecies model is the existence of an error threshold of replication. If replication were error free, no mutants would arise and evolution would stop. On the other hand, evolution would also be impossible if the error rate of replication were too high (since selection would not be able to maintain the genetic information in the population). The notion of error threshold allows us to quantify the resulting minimal replication accuracy that still maintains adaptation.

1.2 Error Thresholds and Optimal Mutation Rates

The notion of error threshold is intuitively related to the idea of an optimal balance between *exploitation* and *exploration* in genetic search. Too low a mutation rate implies too little exploration; in the limit of zero mutation, no new individuals would arise and the search process would stagnate. On the other hand, with an excessively high mutation rate (close to 1.0), the evolutionary process would degenerate into random search with no exploitation of the information acquired in preceding generations.

Any optimal mutation rate must lie between these two extremes, but its precise position will depend on the other evolutionary parameters and the characteristics of the problem at hand. It can, however, be postulated that a mutation rate close to the error threshold would be optimal for the problem under study, because it would maximise the search done through mutation subject to the constraint of not losing information already gained.

Some biological evidence supports the idea that evolution is effective close to the error threshold; certain viruses (such as the HIV virus), which are very efficient evolving entities, seem to operate very close to their error threshold (Nowak & Schuster, 1992; Bonhoeffer & Stadler, 1993). Moreover, the existence of a relationship between error thresholds and optimal mutation rates has been suggested before in the evolutionary computation community (Hesser & Männer, 1991; Kauffman, 1993). Neither of these works, however, confirm the existence of such a relationship, nor explore the relevance of the notion of error threshold in the context of genetic algorithms.

1.3 Aims

The purpose of this thesis is to bring the notion of error thresholds from the field of molecular evolution to the field of genetic algorithms, and to establish the relevance of this notion in the context of GAs. More precisely, the aims of this work are the following:

- To establish whether the phenomenon of an error threshold can be observed in populations of bit strings evolving under a GA
- To relate error thresholds to the more familiar notion of optimal mutation rates in GAs
- To propose general principles for setting near-optimal evolutionary parameters in GAs in the light of this new knowledge

To achieve this, it is necessary to:

- Estimate both error thresholds and optimal mutation rates on a wide range of landscape structures including real-world domains, and compare these two measures against each other.

- Study the effect of modifying other evolutionary parameters (such as string length, recombination, selection pressure, population size, population replacement, and elitism) on both error thresholds and optimal mutation rates.

1.4 Organisation

This dissertation is organised as follows:

Chapter 2 introduces the general field of evolutionary computation, and describes in detail the most widely known of its approaches: genetic algorithms (GAs). The different components and variants of GAs are discussed, revealing the GA as a family of algorithms rather than a single algorithm. The chapter also discusses the many decisions involved when designing a GA. Among such decisions, parameter setting is discussed in more detail, and a classification of approaches to parameter setting is proposed. Also, a detailed review of approaches so far for effective setting of the mutation rate is presented.

Chapter 3 discusses the notion of *fitness landscapes* and presents some properties of landscapes that are known to have an influence on evolutionary search. It also describes the families of abstract fitness landscapes and real-world domains that are used as test problems throughout this dissertation.

Chapter 4 surveys relevant knowledge about quasispecies and error thresholds from molecular evolution. A preliminary empirical study demonstrating the existence of error thresholds in GAs evolving on simple landscapes is presented. This study reproduces experiments from a molecular biology paper (Boerlijst et al., 1996), but uses a GA instead of the quasispecies model as the underlying model of evolution. Some additional experiments using sexual selection, not included in (Boerlijst et al., 1996), are presented. In particular, assortative mating (preference for similar organisms) and dissortative mating (preference for dissimilar mates) are studied.

Chapter 5 introduces *consensus sequence*[2] plots. These plots, borrowed and adapted from molecular biology, are new to the genetic algorithms community. They constitute an empirical approach for locating error thresholds on complex landscapes. Consensus sequence plots are then used to study the effect of modifying various evolutionary parameters on the magnitude of error thresholds. Specifically, genotype length, selection pressure, population size, elitism, steady-state population replacement, recombination, and assortative mating, are considered. Thereafter, the occurrence of error thresholds is investigated on a wide range of landscape structures: from smooth to very rugged, and from abstract landscapes to real-world domains. The existence and characteristics of the error threshold are shown to depend on the fitness landscape structure. Hence, it is postulated that consensus sequence plots may serve as a tool for visualising the structure of a fitness landscape.

Chapter 6 explores the postulated relationship between error thresholds and optimal mutation rates. The correlation between these two measures is assessed by comparing error thresholds (as estimated in Chapter 5) with optimal mutation rates (as estimated in this chapter) on both abstract landscapes and real-world domains. The effect of modifying various evolutionary parameters on the magnitude of optimal mutation rates is also studied. Finally, optimal mutation rates are investigated on a wide range of landscape structures.

[2] The term *sequence* is in this thesis interchangeable with string or genotype.

Chapter 7 presents a summary, and discusses the main contributions and limitations of this dissertation. Some suggestions for further research are also discussed.

1.5 Published Work

Some of the work contained in this dissertation has previously been published elsewhere, specifically:

- Most of Chapter 4 has been published as:

  Ochoa, G., Harvey, I. (1998) Recombination and Error Thresholds in Finite Populations. *Foundations of Genetic Algorithms 5*, pp. 245–264.

- Parts of Chapters 5 and 6 appeared in:

  Ochoa, G., Harvey, I., Buxton, H. (1999). On Recombination and Optimal Mutation Rates. *Proceedings of the Genetic and Evolutionary Computation Conference*, Vol. 1, pp. 488–495.

  Ochoa, G., Harvey, I., Buxton, H. (1999). Error Thresholds and their Relation to Optimal Mutation Rates. *Proceedings of the 5th European Conference on Advances in Artificial Life*, LNAI 1674, pp. 54–63.

  Ochoa, G., Harvey, I., Buxton, H. (2000). Optimal Mutation Rates and Selection Pressure in Genetic Algorithms. *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 315–322.

  Ochoa, G. (2000). Consensus Sequence Plots and Error Thresholds: Tools for Visualising the Structure of Fitness Landscapes. *Parallel Problem Solving from Nature VI*, pp. 129–138.

Chapter 2

Genetic Algorithms: Algorithm Design and Parameter Setting

Evolutionary Computation embraces computer-based search methods inspired by the mechanisms of natural evolution. Several approaches to evolutionary computation have been proposed, which are generally referred to as evolutionary algorithms (EAs). Among these approaches, genetic algorithms (GAs) are probably the most widely known. The first part of this chapter introduces GAs and describes in detail their major components (Section 2.2). It also discusses the informal, but widely used, notion of *selection pressure* of an evolutionary algorithm (Section 2.3).

Given the diversity and possible parameterisations of the various GA components, the designer faces a large number of choices about how to proceed when applying a GA to a given problem. The second part of this chapter discusses such choices, with an emphasis on how to set the various evolutionary parameters (Section 2.5). A classification of approaches to GA parameter setting is proposed (Section 2.5.1). Then, a critical review on optimal settings for the mutation rate is presented, following the proposed classification (Section 2.6).

2.1 Evolutionary Computation

Evolutionary Computation (EC) models natural evolution in the design and implementation of computer-based problem solving tools (Spears et al., 1993). Over the last few decades, several EC models have been proposed and studied, they are collectively referred to as *Evolutionary Algorithms* (EAs) and share the conceptual framework of simulating natural evolution. The theory of evolution was proposed in the 19th-century by Charles Darwin who provided a scientific explanation, essentially correct but incomplete, of how evolution occurs. Natural selection was the fundamental concept in his explanation. Later, work on *genetics*, a science born in the 20th century, revealed in detail how natural selection works and led to the development of the modern theory of evolution, also called *neo-Darwinian theory*. This modern theory can be summarised in the following six propositions (Patterson, 1999):

1. **Reproduction**: 'Like begets like', reproduction in a population of organisms produces descendent populations of similar organisms.

2. **Excess**: The reproductive potential of the parent population always greatly exceeds the actual number of its descendants.
3. **Variation**: Members of a population always vary. Much of this variation is transmitted to the descendants (heritable), and novelties (mutations) may appear.
4. **Environmental Selection**: The space and resources of the environment are limited, so that there is a competition within and between populations. Individuals possessing favourable characteristics, of whatever sort, will tend to compete successfully and leave more descendants than other, less lucky individuals.
5. **Divergence**: The environment varies with time and from place to place. Heritable variations that suit a particular environment will be selected there, and so populations will diverge and differentiate as each becomes adapted to its own conditions.
6. **Common Ancestry**: The principle of divergence has no limit, and the diversity of life on Earth can be explained by divergent descendent lineages from more or less remote common ancestors.

Evolutionary Algorithms mimic natural evolution. Specifically, they model the first four propositions discussed above using algorithmic analogies and the following elements:

- A representation of candidate solutions to the problem at hand
- A population of these candidate solutions
- Mechanisms for generating new solutions from members of the current population (operators such mutation and recombination)
- An evaluation or fitness function to assess the quality (fitness) of a given solution
- A selection method which gives better chances of survival to good solutions

Figure 2.1 outlines a typical evolutionary algorithm. A population of M individuals is initialised and then evolved from generation t to generation $t+1$ by successive applications of fitness evaluation, selection, recombination and mutation.

Historically, there have been three well-defined approaches to evolutionary computation: "evolutionary programming" (Fogel et al., 1966), "evolution strategies" (Rechenberg, 1973), and "genetic algorithms" (Holland, 1975). Although similar at the highest level, these approaches differ in the way they implement an EA. The differences touch all the aspects of EAs, including the choices of representation for the individual structures, types of selection mechanisms, forms of variation operations, and measures of performance. The major differences are, however, in the choice of representations, and emphasis and use of variation operators. Evolutionary programming (EP) uses representations that are tailored to the problem domain. Similarly, evolutionary strategies (ES), due to initial interest in hydrodynamic optimisation problems, use real-valued vector representations. On the other hand, genetic algorithms have traditionally used a more domain independent representation, namely, binary strings. Regarding variation operations, both EP and ES use mutation as the main operator, and propose a form of self-adaptive mutation; whereas GAs emphasise recombination as the main search operator, and use mutation as a secondary operator applied with a small constant probability.

```
Procedure EA {
    t = 0;   /* Initial Generation */
    initialise population(t);
    evaluate(t);
    until (done) {
        t = t+1;
        parent_selection(t);
        recombine(t);
        mutate(t);
        evaluate(t);
        select_survivors(t);
    }
}
```

Figure 2.1: The outline of an evolutionary algorithm.

Although it is not possible to present here a thorough overview of all variants of evolutionary algorithms; it is worth mentioning: *order-based genetic algorithms* (Goldberg, 1989), *classifier systems* (Holland, 1986; Goldberg, 1989), and *genetic programming* (Koza, 1992; Kinnear, Jr., 1997), as branches of genetic algorithms that have developed into their own directions of research and application. Order-based GAs are used in combinatorial optimisation problems where the search space is the space of permutations (e.g. the travelling salesman problem); they work directly on the permutation, applying specialised genetic operators (e.g. inversion and reordering) that preserve permutations. Classifier systems use an evolutionary algorithm to search the space of production rules of a learning system that can induce and generalise. Genetic programming applies evolutionary search to the space of computer programs in a language suitable for modification by mutation and recombination. The dominant approach uses the LISP programming language, but other languages including machine code have also been used (Kinnear, Jr., 1997).

In the last few years, there has been widespread interaction among researchers studying various evolutionary computation methods, and the boundaries between GAs, EP, ES, and other approaches have disappeared to some extent. Nowadays the term "genetic algorithm" is often used to describe a method far from its original definition, which may use other representations than bit strings. This thesis, however, adheres to the traditional binary string representation, and concentrates on the GA approach to evolutionary computation.

2.2 Genetic Algorithms

GAs were developed by John Holland, who summarised his work on adaptive plans in a book entitled "Adaptation in Natural and Artificial Systems" (Holland, 1975). Holland was interested in a general theory of adaptive systems rather than practical applications; his book, however, constituted the starting point of all known GA applications. The major classic applications of GAs are search, optimisation, and machine learning (Goldberg, 1989). There is, however, an increasing recognition that GAs provide a tool in areas where standard approaches fail. The current range of successful applications is broad and touches fields as diverse as engineering, natural sciences, medicine, economics, and business.

The following subsections describe the basic GA components and variants.

2.2.1 Representation or Coding

It is assumed that a potential solution to a problem may be represented as a set of parameters (known as *genes*). These parameters are joined together to form a string of values (referred to as a *chromosome* or *genotype*). Traditionally chromosomes in a GA population take the form of bit strings. However, several other genetic representations have been implemented with success.

2.2.2 Fitness Function

A fitness function must be devised for each problem to be solved. The purpose of the fitness function is to provide a measure of the quality of a candidate solution or chromosome. For many problems (e.g. function and combinatorial optimisation), devising a suitable fitness function is straightforward, but this is not always the case for real-world applications.

2.2.3 Genetic Operators

Genetic operators introduce diversity in the population; they create new individuals from structures in the current population. In GAs, there are two main types of genetic operators: mutation and recombination, which roughly resemble mechanisms in natural asexual and sexual reproduction respectively. Each operator has an associated parameter that controls the probability of its application. How to set these operator parameters is still a matter of discussion in the field. Section 2.5 summarises research work on GA parameter settings.

Mutation

Mutation of a bit involves flipping it: changing a 0 to 1 or vice versa. The probability that a bit will be flipped is given by a parameter (the mutation rate). The bits of a string are independently mutated — that is, the mutation of a bit does not affect the probability of mutation of other bits. Traditionally, mutation in GAs is considered to be a secondary operator whose role is to restore lost genetic material. Some researchers claim, however, that a mutation-selection method constitutes a powerful search algorithm, and that the importance of mutation in GAs has been underestimated while the role of recombination has been overestimated (Schaffer & Eshelman, 1991; Fogel, 1995; Bäck, 1996).

Recombination

Recombination or crossover is considered the main search operator in GAs. This operator produces offspring by merging portions of two selected parents. The idea behind recombination is that segments from different parents should be combined in order to produce new individuals that benefit from advantageous bit combinations of both parents. The application of recombination is controlled by a parameter (the recombination rate). Several recombination operators have been proposed. The most widely known are one-point, multi-point, and uniform crossover. In one-point recombination, a single cut-point is randomly selected within the two parents; then the segments before the cut-points are swapped over. Multi-point recombination is a generalisation of this idea, introducing a higher number of cut-points. Information is then swapped between pairs of cut-points. In uniform crossover (Syswerda, 1989), exchanged segments reduce to single bits, cut-points are not used, instead a global parameter indicates the probability of exchanging each bit between the two parents. Considerable work has been done in comparing recombination operators, but there is no conclusive agreement on what is best. It is likely that the right choice would be problem-dependent. A consensus seems to be, however, that two-point and uniform recombination are generally preferable to one-point recombination.

2.2.4 Selection

Selection allocates reproductive opportunities for each organism in the population. The fitter the organism, the more times it is likely to be selected for reproduction. Selection has to be balanced with variation from mutation and recombination — the *exploitation-exploration* balance. Any efficient optimisation algorithm must balance these two contradictory forces: *exploration* to investigate new areas in the search space, and *exploitation* to make use of information gained so far to find better solutions. Selection in GAs is the component mainly determining the character of the search process; too-strong selection means that suboptimal highly fit individuals will take over the population, reducing the diversity required for further change and progress, whereas too-weak selection will result in very slow evolution.

Numerous selection schemes have been proposed in the literature. However, there are no conclusive guidelines as to which method should be preferred; this is still an open question for GAs. The following subsections describe the most commonly used selection methods.

Proportional Selection

Fitness proportional selection is the classic GA selection mechanism. In this mechanism the reproductive opportunities of an individual is given by its fitness divided by the average fitness of the population. The most common method for implementing proportional selection is the so-called *roulette wheel* – a stochastic method for producing the expected number of offspring for each individual in the population. However, with the population sizes typically used in GAs, the actual number of offspring allocated using the roulette wheel method is often far from its expected value. To minimise these sampling errors, Baker (1987) proposed a new method called *stochastic universal sampling* (SUS). SUS is also more computationally efficient and is chosen in most modern implementations. Thus, it is used in the experiments of this thesis.

Scaling Methods

Scaling the objective function values is a widely accepted practice in GAs. This is done for two reasons. First, to map objective function values to positive numbers, since the standard GA selection mechanism (fitness proportional selection) requires positive fitness values and a maximisation problem. Second, to keep appropriate levels of competition among the individuals throughout a GA run.

Several scaling methods have been proposed, ranging from simple linear transformation to methods that consider some population measures (e.g. fitness standard deviation) for performing an appropriate mapping (see (Goldberg, 1989; Mitchell, 1996) for an overview on scaling mechanisms). But again, there are no rigorous guidelines as to which method should be preferred.

Rank Selection

In rank selection, individuals in the population are ranked according to fitness. The expected number of offspring of each individual depends on its rank rather than on its absolute fitness. There is no need for scaling in this case, since absolute fitness values are not considered. The *linear ranking* method proposed by Baker (1985) works as follows: individuals in the population are ranked in increasing order of fitness, from 1 to M (the population size). The user chooses the expected number of offspring (or expected value) Max ($Max \geq 1$) of the individual with rank M. The expected value of each individual i in the population at time t is given by:

$$ExpVal(i,t) = Min + (Max - Min)\frac{rank(i,t) - 1}{M - 1} \tag{2.1}$$

where Min is the expected value of the individual with rank 1. Given the constraints $Max \geq 1$ and $\sum_i ExpVal(i,t) = M$ (since population size is constant from generation to generation), it is required that $1 \leq Max \leq 2$ and $Min = 2 - Max$. At each generation, individuals in the population are ranked and expected values are assigned according to Equation 2.1. Baker (1985) recommended $Max = 1.1$ and showed that this scheme compared favourably to proportional selection on some selected test problems.

Tournament Selection

In tournament selection, n individuals are chosen at random from the population. The fittest of these individuals is selected for reproduction. Then, all are returned to the original population and can be selected again. This process is repeated as often as necessary to fill the new population. A common tournament size is $n = 2$ (binary tournaments). Tournament selection is similar to rank selection in terms of *selection pressure* (see Section 2.3), but is computationally more efficient and more amenable to parallel implementation.

$(\mu + \lambda)-$ and $(\mu, \lambda)-$Selection

The $(\mu + \lambda)-$ and $(\mu, \lambda)-$selection mechanisms come from the evolution strategies community, but they have been tested in the context of GAs (Bäck, 1991, 1992, 1996). These two methods differ from the standard GA selection mechanisms in that:

- Offspring and parent populations may have different sizes (μ = number of parents, λ = number of offspring).
- Both methods are completely deterministic, there are no selection probabilities.
- Both methods definitely exclude the worst individuals in the population rather than sampling them with small probability (Bäck, 1996).

In (μ, λ)-selection, the μ best individuals out of the λ offspring are selected to become parents of the next generation, while in $(\mu + \lambda)$-selection the μ best individuals are selected from the set of μ parents **and** λ offspring. Thus, the $(\mu + \lambda)$ scheme is elitist (see Section 2.2.6) since it will only accept improvements; it may seem more effective at first glance because it guarantees the survival of the fittest individuals, but it has several disadvantages when compared to (μ, λ)-selection. Particularly, it is not well suited for optimising multimodal functions and for achieving *self-adaptation* of the mutation rates. Thus, (μ, λ)-selection is generally recommended (Bäck, 1996).

2.2.5 Population Replacement

Two basic models of population replacement may be distinguished in GAs: the *generational* model where the whole population is replaced in each generation, and the *steady-state* model where only a few individuals (typically one or two) are replaced in each generation. In between these two extremes a *generation gap* (DeJong, 1975) may be defined as the proportion of individuals which are replaced in each generation. Most GA implementations have used a generational model; this approach is supported by the work of Grefenstette (1986). A more recent trend, however, has favoured steady-state replacement (Whitley, 1989; Davis, 1991). In the steady-state approach, two choices have to be considered. First, how to select two individuals to be parents; and second, how to select one or two unlucky individuals from the population to be killed off. This can be done in several ways, including:

1. Selection of parents according to fitness, and selection of replacements at random
2. Selection of parents at random, and selection of replacement by inverse fitness
3. Selection of both parents and replacements according to fitness/inverse fitness

For example, Whitley's GENITOR algorithm, select parents according to their ranked fitness values, and the offspring replace the two worst members of the population (Whitley, 1989).

The main difference between a generational GA and a steady-state GA is that, in the latter, population statistics (such as average fitness) are computed after each mating, and the new offspring are immediately available for reproduction. Such a GA therefore has the opportunity to exploit a promising individual as soon as it is created. However, Goldberg and Deb (1991) found that the advantages claimed for steady-state replacement are related to an initial growth rate in performance. According to them, the same effect could be obtained by increasing the selection pressure (e.g. using exponential fitness ranking, or large tournament sizes in tournament selection). They found no evidence that steady-state replacement is fundamentally better than generational.

This thesis uses a generational GA as the default approach, but steady-state GAs are also tested to explore the effect of population replacement on both error threshold and optimal mutation rates.

2.2.6 Elitism

The term *elitism*, first introduced by DeJong (1975), describes the idea of retaining the best or some of the best individuals at each generation. In generational GAs, elitism is explicitly implemented by copying the best individual from generation to generation. In steady-state GAs an implicit elitism is achieved if only the least fit individuals are selected for replacement.

Elitism is widely used in practice, and it may seem more effective at first glance because it guarantees the survival of the fittest individuals, however, a non-elitist strategy that allows temporary deterioration to be accepted may help to leave the region of attraction of a local optimum and reach a better optimum.

The experiments in this dissertation use non-elitist GAs as a default, but the effect of elitism on both error thresholds and optimal mutation rates is also explored.

2.2.7 Termination Criteria

Traditionally, a GA run finishes after a fixed number of generations. Often, a fixed number of function evaluations is considered instead of a fixed number of generations. This is a sensible choice because function evaluation is generally the most computationally expensive task of a GA. Moreover, this approach allows fair comparisons between generational and steady-state GAs, and between GAs in general and other search methods, to be performed. Sometimes, termination is controlled by a genotype diversity measure which gauges the average convergence of individuals in the population. Other termination criteria commonly used are: (i) simply that the optimum is reached (this is case with abstract test functions used in empirical studies), (ii) fixed computational time and, (iii) after a number of generations without an improvement.

2.2.8 Performance Measures

Given that GAs are stochastic methods, conclusions can never be drawn from a single run. Instead, statistics (e.g. average, median) from a sufficiently large number of independent runs should be considered. Thus, the standard performance measures for GAs are the average and best fitness values averaged over several runs. Within a given run, the best fitness could be either the current best in the population, or the best fitness attained so far. These measures are considered after a fixed termination criterion, or over fixed intervals throughout the GA run.

DeJong (1975) devised two measures to quantify the effectiveness of different GAs, one to gauge ongoing performance, and the other to gauge convergence to an optimal solution. He called these measures on-line (ongoing) and off-line (convergence) performance respectively. The on-line performance at time t is the average fitness of all function evaluations up to and including time t. The off-line performance at time t, is the average, over t steps, of the best fitness value at each step.

Another GA performance measure is the number of generations or function evaluations required before the GA finds an acceptable solution (or the global optimum if it is known beforehand). The goal is to minimise the number of generations or function evaluations required for finding the solution. The number of evaluations is often preferred as a measure (over the number of generations), because in almost all GA applications the function evaluations dominate execution time.

Finally, in his dissertation, Spears (1998) comments that, although it is a common practice to run GAs to some termination criteria and then report results only after termination, this approach ignores the dynamic aspects of GAs and can lead to overly general conclusions. Conclusions can be surprisingly dependent on the termination criteria, often reversing if a different cut-off is used. Thus, it is a good practice to always show results over the whole run time of a GA.

2.3 Selection Pressure

Selection or selective pressure is an informal term widely used in the GA community to indicate the strength of a selection mechanism. Loosely, the selection pressure indicates the ratio of maximum to average fitness (or expected selection values) in the population. In an attempt to formalise the notion of selection pressure, Goldberg and Deb (1991) introduced the idea of *takeover* time. This approach reflects the effect of selection in the absence of any genetic operator (such as mutation or recombination). The idea is to count the number of generations needed to produce a population consisting completely of the best individual in the initial population. The intuition behind this is that small takeover times characterise strong selective pressure, which corresponds to exploitative search, whereas large takeover times characterise weak selective pressure, corresponding to explorative search.

Bäck (1996) analysed four selection mechanisms (proportional selection, rank selection, tournament selection and (μ,λ)-selection) considering takeover times. He ordered these selection mechanisms according to increasing selection pressure, in particular when "standard" values of control parameters were assumed[1]. The ordering is as follows:

1. Proportional selection
2. Linear ranking
3. Tournament selection
4. (μ,λ)-selection

Selection mechanisms 2, 3 and 4, are based on rank rather than on raw fitness values. According to Bäck's analysis, there is a strong difference between proportional selection, having a takeover time of order $O(M \ln M)$ (where M is the population size), and rank-based methods with a general takeover time of order $O(\ln M)$, that is, a factor of M faster. Moreover, rank-based mechanisms allow explicit control over the selection pressure. Each of them has a single control parameter that can be tuned for varying the strength of selection. For example, in tournament selection, a common tournament size is 2, but selective pressure increases steadily for growing tournament sizes.

2.4 GA Design

In view of the various GA selection mechanisms, population replacement approaches, possible choice of operators, and ranges for evolutionary parameters; the GA is not a single algorithm but rather a class of related algorithms. Moreover, there are actually as many different GAs as there are GA projects. To complicate matters further, there is little (if any) theoretical guidance, and few rules of thumb to assist the user in the design of an evolutionary approach to a given problem.

When applying an evolutionary algorithm to a given problem, two major steps are needed: (i) selecting an adequate representation, and (ii) designing and implementing a fitness function. These two elements form the bridge between the problem context and the algorithm framework. If the selected algorithm is the GA, these further decisions have to be made:

1. Selection method: how to perform selection
2. Choice of operators: what genetic operators to use
3. Parameter settings: how to set the values for the various parameters

The issues addressed in this dissertation are mainly related to the third choice, namely, how to set the values for the various evolutionary parameters. Therefore, the next section discusses this in more detail.

[1]The standard control parameters of selection mechanisms are: for rank selection ($Max = 1.1$), for tournament selection (tournament size = 2), and for (μ,λ)-selection ($\mu/\lambda \approx 7$).

2.5 GA Parameter Setting

The values of the evolutionary parameters strongly determine the performance of the algorithm. Specifically, they determine whether the algorithm will find a near-optimal solution and whether it will find such a solution efficiently. But finding a good combination of parameters is not an easy task. The evolutionary parameters interact with one another non-linearly, thus they cannot be optimised one at time. Moreover, the 'optimal' parameter setting is likely to depend on the problem at hand.

During the 70's and 80's, a standard GA using bit strings, one-point recombination, bit-flip mutation, and roulette wheel selection (with or without elitism) was widely used. Algorithm design was thus limited to choosing the so-called control parameters, such as population size, mutation rate (per bit), and recombination rate. Most researchers based their choices on "tuning" the parameters by trial and error, that is, experimenting with different values and selecting those producing the best results.

At that time, three major empirical studies attempted to provide a good combination of parameter values. The first study (DeJong, 1975), proposed a test suite of five functions and studied the on-line and off-line performance (defined in Section 2.2.8) on them. His results suggested the following parameter values:

- population size: 50 - 100
- crossover rate: 0.6
- mutation rate (per bit): 0.001

These settings (along with De Jong's test suite) became widely used in the GA community, even though it was not clear how well they would perform on problems outside De Jong's test suite.

Ten years later, Grefenstette (1986) suggested a different approach, he used a "meta-GA" to evolve parameter combinations for the problems in De Jong's test suite. This method produced an interesting parameter combination which significantly increased the on-line performance, but was unable to outperform De Jong's values for off-line performance. Grefenstette recommended values were (values to optimise the off-line performance are given in parentheses):

- population size: 30 (80)
- crossover rate: 0.95 (0.45)
- mutation rate (per bit): 0.01 (0.01)

Notice that Grefenstette's results suggest a smaller population size and higher operator probabilities than De Jong's. This was an interesting experiment, but again, in view of the specialised test suite, it is not clear how generally these recommendations hold.

Later on, Schaffer et al. (1989) carried out extensive studies with high CPU time, to explore a wide range of parameter combinations. They used the on-line measure to gauge GA performance, expanded De Jong's suite with new five functions, and employed Gray code[2] to represent variables. Schaffer et al. found that the best settings for population size, crossover rate, and mutation rate

[2]A Gray code represents each number in the sequence of integers $\{0...2^{L-1}\}$ as a binary string of length L in an order such that adjacent integers have Gray code representations that differ in only one bit position.

were independent of the problem in their test suite. These settings were similar to those found by Grefenstette:

- population size: 20 - 30
- crossover rate: 0.75 - 0.95
- mutation rate (per bit): 0.005 - 0.01

Notice that the purpose of the three approaches described above was to find an effective and general combination of parameters. In other words, the underlying assumption was that the recommended values can be applied to a wide range of optimisation problems. Formerly, GAs were seen as robust problem solvers that exhibit approximately the same behaviour over a wide range of problems. However, the contemporary view on evolutionary computation holds that specific problems (problem types) require specific algorithm setups for satisfactory performance (Eiben et al., 1999).

To summarise, it seems very difficult to formulate *a priori* general principles about parameter settings, in view of the variety of problem types, encodings, and performance criteria that are possible in different applications. Moreover, it has been suggested that the optimal population size, recombination rate, and mutation rate are likely to change over the course of a single run (Mitchell, 1996). Many researchers in the evolutionary computation community consider that the most promising approach is to *adapt* the parameter values in real time through the ongoing search. This has long been the approach in the evolution strategies community. There have also been several approaches to adaptation of evolutionary parameters in GAs. Before discussing them, let us first present a global taxonomy of parameter settings in EAs inspired by the classification of Eiben et al. (1999).

2.5.1 Classification of Approaches to Parameter Setting

Eiben et al. (1999) distinguish two major forms of setting evolutionary parameters: parameter *tuning* and parameter *control*. By parameter tuning the authors mean the common practice of somehow finding good parameter values before the run, and then running the algorithm using these values, which remain fixed during the run. By parameter control, they refer to the alternative of starting a run with initial parameter values which are changed during the run.

Eiben et al. (1999) further categorise the approaches to parameter control according to two aspects, namely, the type of update mechanism, and the EA component subject to changes. They distinguish three types of update mechanisms: *deterministic*, *adaptive*, and *self-adaptive*; and the following EA components: representation, fitness function, operators and their probabilities, selection method, population replacement, and population size. The following describes the three types of approach for changing the value of a parameter (or update mechanism):

- **Deterministic**: When the value of a parameter is altered by some deterministic rule. This rule modifies the parameter without using any feedback from the search. Usually, a time-varying schedule is used, i.e. the rule is applied after a predefined number of generations since the last rule activation.

- **Adaptive**: When some feedback from the search is used to determine the direction and/or magnitude of the parameter change.

- **Self-Adaptive**: Here the parameters to be adapted are encoded into the genotypes and undergo evolution through mutation and recombination. The idea is that better values of the encoded parameters lead to better individuals, which in turn are more likely to survive and reproduce and hence, propagate these better parameter values.

I follow here the classification discussed above but with some modifications and extensions. The terms parameter *tuning* and *control* are substituted by *static* and *dynamic* parameter setting respectively. These latter terms, in my opinion, better capture the essential difference between the two approaches. Moreover, I suggest that static parameter setting should be divided into two categories, namely, parameter *tuning* and parameter *heuristics*. By parameter tuning, I understand the common practice of finding good parameter values for a given problem by trial and error, whereas parameter heuristics refer to rules of thumb that have wider applicability. Static parameter heuristics and dynamic deterministic setting, may also be further divided into empirical and theoretical approaches. Figure 2.2 shows both the global taxonomy of parameter setting as suggested by Eiben et al. (1999) (left), and the modified version proposed in this section (right).

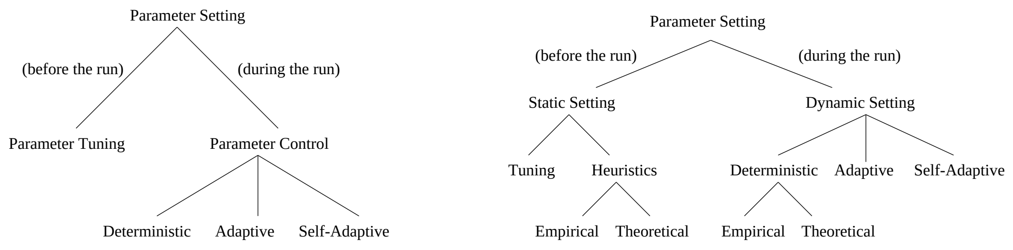


Figure 2.2: Global taxonomy of parameter setting in EAs. Left, classification suggested by Eiben et al.(1999). Right, modified version proposed in this section.

2.6 Optimal Mutation Rates

It has been suggested that the most sensitive of GA parameters is the mutation rate (Schaffer et al., 1989; Bäck, 1996). Moreover, mutation rates are the main subject of this dissertation. Therefore, this section presents a critical review of approaches for 'optimal' setting of the mutation rate according to the classification proposed above (Figure 2.2, right).

2.6.1 Static Setting

Tuning

By tuning I refer to the common practice of finding the evolutionary parameters 'by hand', that is, by experimenting with different values and selecting the ones producing best results. When using this approach, researchers report the parameter values used together with their results, without much justification of the choices made.

Heuristics

Several authors have tried to find useful heuristics for setting 'optimal' mutation rates. These attempts may be divided into empirical and theoretical.

- **Empirical Approaches**: Recapitulating from the empirical approaches described in Section 2.5, the values proposed for the mutation rate (per bit) were: $p_m = 0.001$ (DeJong, 1975),

$p_m = 0.01$ (Grefenstette, 1986), and $p_m \in [0.005, 0.01]$ (Schaffer et al., 1989). Schaffer et al. (1989) also arrived at an empirical expression by curve fitting of their data (M is the population size and L the chromosome length):

$$p_m \approx \frac{1.75}{M\sqrt{L}}, \tag{2.2}$$

The limitation of these early approaches lies in the specialised test suites used (from standard function optimisation problems). It is not clear how the GA will perform with these settings outside the particular test suites used. Others have found fitness functions for which these mutation rate values are not optimal (e.g. Smith and Fogarty (1996)).

- **Theoretical Approaches**: Mühlenbein (1992), following earlier work by Bremerman et al. (1966), theoretically analysed optimal mutation rates for a simple asexual GA with population size 1 on the simple *Onemax* function[3]. Using an approximation of the probability for improving the fitness function by mutation, the author arrived at an optimal mutation rate (per bit) $p_m = 1/L$, where L is the string length. Given the extremely simplified algorithm and fitness function, the practical relevance of this result is questionable. However, the heuristic of $p_m = 1/L$ has produced surprisingly good results in practice, and will be discussed later on (Section 2.6.3).

2.6.2 Dynamic Setting

Deterministic

These approaches alter the value of a parameter by some deterministic rule. Deterministic dynamic approaches may also be categorised into empirical and theoretical.

- **Empirical Approaches**: Fogarty (1989) empirically examined the effect of varying the mutation rate over time and across the bit representation of individuals. He found that varying the mutation rate in either or both of these ways significantly improved the GA performance in the problem studied (an industrial application) but only when starting from a conservative initial population of all zeros (whereas the standard practice in GAs is to start from a randomly initialised population). This study, however, makes an important contribution, since it was the first time that the mutation rate was changed during the run of a GA.

- **Theoretical Approaches**: Hesser and Männer (1991) theoretically analysed the dependence of the mutation rate upon both the population size and the chromosome length. They used arguments from the theory of GAs, stochastic processes, and theoretical biology. The authors introduced a time-dependency into the mutation rate, confirming the findings of Fogarty described above. They proposed an expression for optimal setting of the mutation rate (per bit) for a special GA-variant on the Onemax problem:

$$p_m(t) = \sqrt{\frac{\alpha}{\beta}} * \frac{exp(-\gamma\frac{t}{2})}{M\sqrt{L}} \tag{2.3}$$

where α, β, γ are constants tied to the particular fitness function, and M and L are the population size and the string length. Notice that this expression is similar to that proposed by Schaffer et al. (Equation 2.2), in that the optimal mutation rate is inversely proportional to both the square root of the string length and the population size. The practical relevance of this study is questionable given that a special GA and a very simple function were used. However, the idea of a using a time-varying mutation rate was again proved useful.

[3] The Onemax or counting-ones function gives the number of 1s in a bit string. Thus, the fittest string is the string of all ones.

Bäck (1992) presented an analysis of optimal mutation rates for a simplified GA (no recombination and population size of 1) on the Onemax function. He found that the optimal mutation rate strongly depends on the current Hamming distance to the optimal solution. In other words, the mutation rate should not be constant but should decrease over time during the search. The author presented an approximated schedule for optimal setting of the mutation rate (per bit), in terms of the current fitness value f_a:

$$p_m(f_a) \approx \frac{1}{2(f_a+1)-L} \qquad (2.4)$$

The usefulness of this expression is questionable since the distance to the optimum is not known in real applications. Moreover, this analysis only applies for a simplified GA on the very simple Onemax function.

In a later paper, Bäck and Schütz (1996) proposed a deterministic mutation rate schedule analogous to the expression derived for the Onemax function (Equation 2.4). They used a time-dependent per bit mutation rate $p_m(t)$ where $t \in 0,1,\ldots,T-1$ denotes the generation counter, and T is a given maximum number of generations. From the conditions $p_m(0) = 1/2$ and $p_m(T-1) = 1/L$, their formulation produced:

$$p_m(t) \approx \frac{1}{(2+\frac{L-2}{T-1}*t)} \qquad (2.5)$$

The authors commented that it is not clear whether this substitution of the distance to the optimum by the generation number is useful. But, in their study, they found that the schedule represented by Equation 2.5 outperformed two other approaches for optimal setting of the mutation rate (a self-adaptive approach, and the static heuristic $p_m = 1/L$ per bit) on a test suite of 3 combinatorial optimisation problems (Bäck & Schütz, 1996).

Adaptive

These approaches use some feedback from the search to determine the direction and/or magnitude of the parameter change. The idea of an adaptive control of the mutation rate comes from the evolution strategies community (Rechenberg, 1973). In GAs, similar approaches have been used to adjust both mutation and recombination rates; the idea is to use the quality of the offspring generated by an operator as a measure to adapt the probability of its application. This measure is also called *operator productivity*. The earliest of these techniques was devised by Davis (1989). His method keeps records, for each member of the population, about which operators were used to produce them and their ancestors, and any improvement that the operators were able to attain. Davis showed that this method improved the performance of a GA on some problems. Some years later, Julstrom (1995) proposed a similar technique that requires less bookkeeping. Each member of the population has a tree attached to it depicting the operators used to create it. When a child of improved fitness is produced, this tree is used to assign credit to each operator. Both Davis and Julstrom methods periodically process this information to adjust the operator settings; and both have been observed to effectively adapt the operator settings.

Tuson (1995), Tuson and Ross (1998) object that the above methods require much additional bookkeeping. According to them, it is entirely possible that a simpler approach would work just as well. Hence, they investigated a simpler approach: COBRA (Cost Operator Based Rate Adaptation) (Corne et al., 1994), originally devised for adapting operator settings in time-tabling problems. Initial operator settings are provided and the GA periodically swaps them between operators, giving the highest probability to the operator that has been producing the most gains in fitness. Although COBRA was shown to exhibit improved performance in time-tabling problems, when compared to a similar GA with fixed operator settings (Corne et al., 1994), no improvement in performance was found on the test problems studied by Tuson and Ross (1998). According to the authors, however, the GA was often less sensitive to the initial operator settings when COBRA was used, which in some applications may be useful.

Adaptive approaches are based on the principle that dynamically changing fitnesses of operators should keep up with their actual usefulness at different stages of the search, causing the GA to use them at appropriate rates at different times. According to Mitchell (1996), this ability for the operator fitness to keep up with the actual usefulness of the operators has not been tested in any way. She comments further that a big question for adaptive approaches to setting parameters is:

> How well does the rate of adaptation of parameter settings match the rate of adaptation in the GA population? The feedback for setting parameters comes from the population's success or failure on the fitness function, but it might be difficult for this information to travel fast enough for the parameter settings to stay up to date with the population's current state [(Mitchell, 1996), p. 177].

Self-Adaptive

Here the parameters to be adapted are encoded into the genotypes and undergo evolution through mutation and recombination. Self-adaptive control of mutation step-sizes is traditional in evolution strategies (Bäck, 1996). Several attempts have been made to bring this idea to GAs. The work of Bäck (1991) includes mutation rates a as part of the genetic representation of individuals (encoded as bit-strings). Hence, mutation rates are subject to adaptation as well as the objective variables. Bäck experimented with two types of selection mechanisms: *extinctive selection* and *preservative selection*[4]. As test functions, he selected 3 functions from the classic optimisation test suites. His results suggest that self-adaptation of mutation rates is advantageous and possible for GAs, but only when extinctive selection is used (that is for a very high selection pressure). In this study, however, the author compared the self-adaptive approaches with a standard GA with a per bit mutation rate of $p_m = 0.001$. This value was selected without any particular justification (besides being a common figure in standard GAs). One objection to this empirical comparison is then, that a mutation rate of $p_m = 0.001$ is probably very far from optimal (indeed too low) as a static value for the test suite selected.

Later on, Bäck and Schütz (1996) compared three different approaches for optimal mutation rate setting:

- A constant setting — $p_m = 1/L$ per bit
- A deterministic time-varying mutation rate schedule
- A self-adaptation mechanism

[4]An extinctive selection mechanism definitely excludes some individuals from being selected, whereas preservative mechanisms always assign selection probabilities greater than zero to all individuals. According to Bäck's analysis, extinctive selection imposes a much higher selection pressure.

As test functions, they selected 3 combinatorial optimisation problems. The deterministic mutation rate schedule used was the one described above (Section 2.6.2, Equation 2.5). Regarding the self-adaptation mechanism, the authors proposed a refinement over the approach presented in (Bäck, 1991). They acknowledged that the previous approach was of limited success and postulated that the binary representation of mutation rates used there hampered the efficient fine-tuning by self-adaptation. Thus, they proposed a self-adaptation mechanism where mutation values were encoded as real numbers, and were subject to evolution by mutation only (no recombination). The authors presented a comparison among the three proposed mutation regimes (listed above) on the combinatorial test problems. When comparing the self-adaptive regime with the static regime, results suggest that the self-adaptive mutation rate has better performance on average, but not with respect to the number of times that optimal solution was found. The deterministic control regime was the best of the three, regarding both average final fitness and the number of runs that yield the optimum. It should be noticed, however, that this study used GAs without recombination, and the differences in performance were, in my opinion, not dramatic.

The work of Smith and Fogarty (1996) investigated the use of genetically encoded mutation rates within a steady-state GA. They tested several selection and deletion policies, and included a form of local search in their self-adaptive mechanism. As test problems they used the NK model (described in Chapter 3) with several values of K which covered a range of landscape structures from smooth to very rugged. A comparison between the best self-adaptive GA found, and a GA with standard fixed mutation rates was presented. Four fixed values for the per bit mutation rate were selected for comparison : $p_m = 0.001$ (DeJong, 1975), $p_m = 0.01$ (Grefenstette, 1986), $p_m = 1/L$ (where L is the string length), and $p_m = 1.75/M\sqrt{L}$ (where M is the population size) (Schaffer et al., 1989). Two studies were carried out. The first compared the self-adaptive GA (with local search), against the standard steady-state GA (without local search) with the four selected static mutation values. Results indicated that the self-adaptive GA significantly outperformed the standard steady-state GA for all the fixed mutation values, with $p_m = 1/L$ giving the best results among the static settings. A second (more fair) comparison also included local search on the standard steady-state GA. Again, $p_m = 1/L$ per bit produced the best performance among the static mutation values. In this case, results with $p_m = 1/L$ were similar to the self-adaptive GA on the simplest landscapes, and significantly better for the self-adaptive GA on the more complex landscapes.

A study of self-adaptive parameter settings was also carried out by Tuson and Ross (1998). They used real numbers to encode both mutation and recombination parameters in each individual. Two types of *meta-operators* (i.e. the operators applied to the encoded operator settings) were tested. First, *strongly disruptive* operators, which tend to produce children quite different from their parents; and second, *weakly disruptive* operators, which produce children that are similar to their parents. They found that the choice of these meta-operators had a dramatic effect: disruptive operators were found to remove the ability to adapt as they destroy any information gained by selection. The use of low disruptive operators improved matters somewhat, but the occurrence of adaptation was unreliable and depended on the problem. Not surprisingly, when no adaptation took place, the effect upon performance was often detrimental. Moreover, performance also declined even when adaptation *did* take place. The authors argued that it takes time for operator parameters to evolve to the right values, by which time much of the useful search has already been performed, and, hence, the impact of the evolved settings is much reduced. Thus, setting appropriate operator values at the start of the GA run appears to be important. Tuson and Ross concluded by saying that: "Operator adaptation was not found to be as universally useful as earlier studies on this subject have implied, in that it will not necessarily produce results that are superior to modest hand-tuning ...".

2.6.3 Discussion

All the approaches described above for effectively setting the mutation rate have intrinsic limitations. Hand tuning is time-consuming and often not viable. Theoretical approaches are, in general, of questionable practical relevance since they are based on very simplified algorithms and fitness functions. Early empirical approaches using standard function optimisation test suites, are of restricted generality.

Both adaptation and self-adaptation of operator parameters have a computation overhead. Moreover, the application of these techniques for setting the mutation rate in GAs has shown, in my opinion, only limited success. Since adaptation and self-adaptation are successful within the evolution strategies community, this issue deserves further investigation in the context of GAs.

In my opinion, the most useful guideline so far for an effective and general setting of the mutation rate in GAs is the heuristic suggesting $p_m = 1/L$ (per bit). This figure has appeared several times in the evolutionary computation literature. The earliest appearance I can trace back was due to Bremerman et al. (1966) as quoted by Bäck (1996). Also, DeJong (1975) suggested this value as quoted by Hesser and Männer (1991). The work of Mühlenbein (1992) states that $p_m = 1/L$ is optimal for general unimodal functions. This setting has also produced good results for several NP-hard combinatorial optimisation problems such as the multiple knapsack problem (Khuri, Bäck, & Heitkötter, 1994), the minimum vertex cover problem (Khuri & Bäck, 1994), the maximum independent set problem (Bäck & Khuri, 1994), and others (Bäck & Khuri, 1994). The work of Smith and Fogarty (1996) found $1/L$ as the best fixed setting for the mutation rate, giving results comparable to their best self-adaptive method. Other authors have found a dependence of effective mutation rates upon the string length L, although they had not explicitly suggested $p_m = 1/L$ (Schaffer et al., 1989; Hesser & Männer, 1991; Bäck, 1992, 1993).

Thus, there may well be some true principle underlying this heuristic. It is argued, in this thesis, that this principle is related to the notion of error threshold from molecular evolution. The error threshold is the minimal replication accuracy that still maintains genetic information in the population. Chapter 4 discusses this notion in more detail; and Chapter 6 studies its relationship with the more familiar notion of an optimal mutation rate in GAs. Let us, however, anticipate here that the theoretical expression of the error threshold on a simplified landscape (a single peak landscape) is $p_m = ln(\sigma)/L$ (per bit) (Eigen & Schuster, 1979), where L is the genotype length and σ is the *superiority* parameter of the master sequence. The master sequence is the current fittest sequence in the population, and σ is the factor by which selection of this master sequence exceeds the average selection of the rest of the population; in other words, σ is a measure of the selection pressure. A resemblance between the expression for error thresholds and the heuristics of $p_m = 1/L$ (per bit) is observed if one assumes that the ratio of maximum to average fitness in the population (the selection pressure) is ≈ 2, which is the case for linear ranking (with $Max = 2$, see Equation 2.1), and tournament selection with tournament size of 2.

The idea of optimal mutation rates being related to error thresholds is implicitly suggested by Harvey (1992, 1997), who presents a GA framework (SAGA) specially designed for evolving genetically converged populations of variable length genotypes. In this framework, mutation is considered as the primary search operator. The mutation rate is selected following the expression of error thresholds on a single peak landscape $p_m = ln(\sigma)/L$ (whenever the value of σ may be estimated or approximated). A further adjustment of this guideline is suggested in the presence of redundancy or junk in the genotype. If the target mutation rate is, for example, 1 mutation per genotype on the assumption of no redundancy, in the presence of junk this should be increased so as to give an expected 1 mutation per *non-redundant* part of the genotype (again, whenever the proportion of redundancy may be estimated). This adjustment was empirically validated in the context of evolvable hardware experiments (Harvey & Thompson, 1996).

2.7 Summary

This chapter introduced the general field of evolutionary computation and described in more detail the most widely known of its approaches: genetic algorithms (GAs). The different components and variants of GAs were described, revealing that the GA is a family of algorithms rather than a single algorithm. Indeed, there are as many different GAs as there are GA projects. To complicate matters further, there is little (if any) theoretical guidance, and few rules of thumb to assist the designer when applying a GA to a given problem. Thus, the second part of the chapter discussed the many decisions involved when designing a GA. One of such set of decisions, the parameter settings, was discussed in more detail, and a classification of approaches to parameter setting was proposed. The main concern of this thesis is the mutation rate, so a review of approaches so far for effective setting of the mutation rate was presented. This review was structured according to the suggested classification of approaches.

All the approaches discussed in this chapter for a near-optimal setting of the mutation rate have intrinsic limitations. A promising guideline is, however, the heuristic suggesting $p_m = 1/L$ where L is the string length. This heuristic has appeared several times in the GA literature, has produced good results in practice, and seems to be supported by the notion of error threshold (discussed in detail in Chapters 4 and 5). Bearing in mind the postulated underlying relationship between error thresholds and optimal mutation rates, Chapter 6 of this thesis attempts to assess the generality of the $p_m = 1/L$ heuristic, find further adjustments, and detect the limits of its applicability. First, however, Chapter 3, describes the fitness landscapes used as test problems throughout this dissertation.

Chapter 3

Fitness Landscapes and Test Problems

The notion of fitness landscapes was introduced to describe the dynamics of adaptation in nature (Wright, 1932). Since then, it has become a powerful metaphor in evolutionary theory. Fitness landscapes are also well suited to describe the dynamics of artificial evolution. Hence, evolution (natural or artificial) can be seen as an adaptive-walk over a fitness landscape. Identifying the structure of fitness landscapes may be helpful in both predicting the performance and improving the design of evolutionary algorithms.

This chapter is structured in two main sections. Section 3.1 introduces the notion of fitness landscapes and presents some properties of landscapes that are known to have an influence on evolutionary search. It also discusses briefly some approaches proposed so far for analysing the structure of fitness landscapes. Section 3.2 describes the families of abstract landscapes and real world-domains that will be used as test problems throughout this dissertation. It also justifies this particular choice of test problems. Two real-world domains were selected for study (Section 3.2.2), namely, a combinatorial optimisation problem — the Multiple Knapsack problem, and an engineering problem — the design of an optimal aircraft Wing-Box.

3.1 Fitness Landscapes

The biologist Sewall Wright (1932) envisioned the consequences of natural selection as an adaptive walk over a fitness landscape. Natural organisms can be viewed by their *genotype*, which is the genetic ‘encoding’ of the organism, or their *phenotype*, which is the actual form and behaviour of the organism. An abstract notion of *fitness* can be assigned to each phenotype that measures its ability survive and reproduce. Evolution is then viewed as a process that searches, by means of genetic operators like mutation and recombination, a fitness landscape of possible genotypes, looking for genotypes that encode highly fit phenotypes. Depending on the distribution of phenotypes, the fitness landscape can be more or less hilly. It may have many peaks of high fitness flanked by ridges and cliffs falling to profound valleys of low fitness. Or it may be smooth, with low hills and gentle valleys.

In this framework, adaptive evolution is a hill-climbing process. The population is a tight or loose cluster of individuals located in the landscape. Mutation and recombination move individuals (or their offspring) to neighbouring points in the space. Over time, the cluster of individuals will flow over the fitness landscape. In the simplest cases, the population will climb to and cluster about one of the possible multiple peaks. In more complex cases, the population cluster may spread widely across the landscape, passing through a web of ridges somewhat below the fitness peaks (Kauffman, 1993).

3.1.1 Fitness Landscapes in Sequence Space

Although the notion of fitness landscapes was first introduced in the context of evolution at the level of organisms, it has recently gained relevance in the context of evolution at the molecular level. In molecular evolution the space of all possible configurations can be captured by the notion of *protein space*, or more generally *sequence space* (Maynard Smith, 1970). Molecules (such as proteins and nucleic acids) consist of specific sequences (or strings) drawn from finite alphabets. Consider sequences of letters drawn from an alphabet of A characters. If the sequence is of length L, then there are A^L possible sequences of that length. For example there are 20^L possible proteins of L amino acids and 4^L possible poly-nucleotides of L nucleotides. This means that even for moderate sequence lengths a "hyper-astronomically" large number of different variants can be formed. Let us imagine all possible sequences to be arranged in a sequence space such that two sequences are neighbours if one can be converted into another by a single point mutation. Thus the sequence space is formed by a set of sequences (of uniform length L) together with a definition of distance between sequences. An appropriate definition is given by the Hamming distance [1]. For a binary alphabet, sequence space can be visualised as an L-dimensional hypercube (Figure 3.1). Once we have defined the sequence space, the missing component of a fitness landscape is a mapping from sequences to real numbers. These numbers measure the sequences ability to perform a given function.

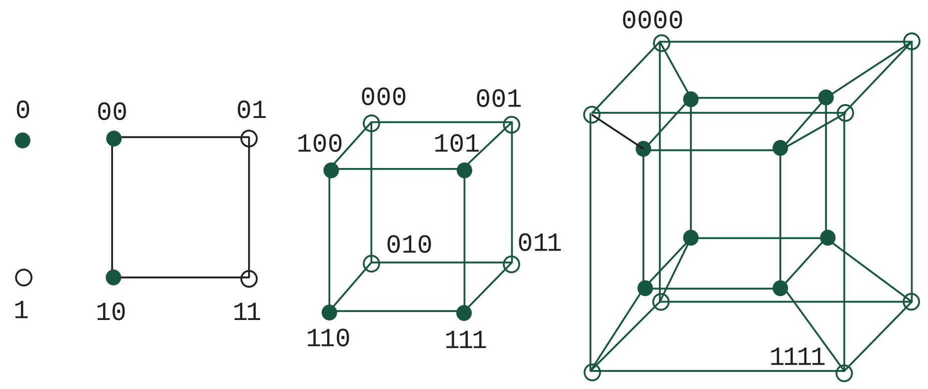


Figure 3.1: Visualising sequence space. Hypercubes for sequence lengths $L = 1$ through $L = 4$. Edges connect sequences of Hamming distance one.

3.1.2 Fitness Landscapes in Evolutionary Computation

The fitness landscape metaphor can be used for search in general. Given an optimisation problem, the set of possible solutions can be coded using strings of (generally) fixed length from some finite

[1] The Hamming distance counts the number of positions where two sequences differ.

alphabet. This encoding generates a *representation space*, which is a high dimensional space of all possible strings of a given length. There is also a *neighbourhood relation* that defines which points in the representation space are connected. This relation depends on the specific search operator (e.g. mutation) or combination of operators (e.g. mutation and recombination), used to search the space. Finally, there is a fitness function that assigns a fitness value to each possible string or point in the space (Hordijk, 1997). To summarise, a fitness landscape is defined by:

1. A representation space (all possible strings in the encoding)
2. A neighbourhood relation denoting which points in the representation space are neighbours
3. A fitness function that assigns a fitness value to each point in the space

3.1.3 Properties of Fitness Landscapes

In the context of evolutionary computation, it is important to identify the features of landscapes that influence the effectiveness of evolutionary search. Such knowledge may be helpful for both predicting the performance and improving the design of EAs. The following properties are known to have a strong influence on evolutionary search (Merz & Freisleben, 1999).

- the fitness differences between neighbouring points in the landscape (*ruggedness*),
- the number of local optima or peaks in the landscape,
- the distribution of the local optima in the search space, and
- the topology of the basins of attraction of the local optima.

An important characteristic of a landscape is its *ruggedness*, which is related to the difficulty of an optimisation problem for evolutionary algorithms. A landscape where nearby points tend to have similar fitness values, is called *smooth*. On such a landscape it will be easy to find good optima: there will be few peaks, and local information about the landscape can be used effectively to direct the search. On the other hand, a landscape where nearby points tend to have dissimilar fitness values, is called *rugged*. On such a landscape it will be difficult to find a good solution: there will be many peaks, and local information becomes less valuable. Hence, the global structure of a landscape can range from very smooth to very rugged.

Selective Neutrality

The view of adaptive evolution as a hill-climbing process has dominated evolutionary biology since the introduction of the fitness landscape metaphor (Wright, 1932). The importance of *selective neutrality* [2] as a factor in evolution has, however, been stressed more recently. In particular from the study of quasispecies (Eigen, 1971; Eigen & Schuster, 1979), and the analysis of RNA evolution (Fontana & Schuster, 1987; Huynen, 1995). Research on the evolution of RNA molecules, both in vitro and through simulation, suggests that the evolutionary process is shaped by a high degree of redundancy in sequence-structure[3] mappings; there are many more sequences

[2]The Neutral theory of evolution suggests that evolution at the molecular level is dominated by non-adaptive neutral changes (Kimura, 1982).

[3]The function of proteins and nucleic acids is determined by their 3D structure. The so called *tertiary structure* refers to the looping and folding of the molecule chain back upon itself.

than structures, and sequences folding into the same structure are (almost) randomly distributed in sequence space. Two consequences of this redundancy are important for evolution. First, the space of possible structures is covered by small connected regions in sequence space. Second, the existence of extended *neutral networks* (i.e. sets of equal-fitness sequences that can reach each other via elementary genetic variation steps such as point mutation). These two consequences explain how evolution in nature can effectively find solutions to complex problems on very large search spaces. In the presence of neutral networks, populations avoid being caught in evolutionary traps and eventually reach the global optimum through a composite dynamics of adaptive walks and random drift (Schuster, 1994). The whole picture can be captured in the term "smoothness within ruggedness", on average the landscape may be very rugged, but there exist paths that percolate through genotype space on which the structure remains unaltered (Huynen, 1995).

To summarise, this subsection concerning the properties of landscapes, distinguishes two important landscapes features, namely, (i) ruggedness, and (ii) neutrality. These features will be considered when selecting the fitness landscapes used as test problems throughout this dissertation (Section 3.2).

3.1.4 Landscape Analysis Techniques

This section briefly describes various techniques for studying the structure of landscapes. The global structure of landscapes can be mathematically expressed by the landscape *correlation structure*, which is determined by the fitness differences between neighbouring points. Small differences give a highly correlated landscape, while large differences give an uncorrelated landscape. In between, there is a whole range of more or less correlated landscapes. From this correlation structure, a *correlation length* can be derived. The correlation length measures the largest "distance" between two points at which the fitness of one point still provides information about the expected fitness of the other point. Several methods for measuring the correlation structure and correlation length of a fitness landscape have been proposed in the literature (Weinberger, 1990; Lipsitch, 1991; Manderick et al., 1991; Hordijk, 1997).

Measuring the correlation characteristics of a fitness landscape is an easy and reliable way to assess its ruggedness. However, the correlation measures provide generalised and often insufficient information about the landscape structure. Vassilev, Fogarty, and Miller (2000) proposed a new *information analysis* of fitness landscapes. They defined and studied three basic information characteristics of landscapes: information content, partial information content, and information stability. This information analysis is different from other statistical approaches, and gives us a notion of the interplay between the smooth, rugged, and flat (neutral) landscape areas.

The following section will describe the landscapes used as test problems in the empirical chapters of this dissertation. A justification of the particular choices of test problems will also be provided.

3.2 Test Problems

This section describes the test problems used in this dissertation. First, a group of abstract fitness landscapes were selected: Royal Staircase functions, NK landscapes, and NK landscapes with neutrality (NKF). This selection is consistent with the belief, held in this dissertation, that ruggedness and neutrality are two important landscape features found in real-world applications. The Royal Staircase family of functions is a very simple class of functions that allows neutrality to be modelled and tuned. The *NK* family of landscapes is a problem-independent model for constructing landscapes that can gradually be tuned from smooth to rugged. Finally, the *NKF* family of landscapes allows both ruggedness and neutrality to be gradually tuned. By using these three abstract models, a wide range of landscape structures can be explored, some of which might share features with landscapes from practical applications.

Second, in order to study whether the issues explored in this dissertation carried over from abstract landscapes to real-world applications, two real-world domains were selected: a combinatorial optimisation problem — the Multiple Knapsack problem, and an engineering problem — the design of an optimal aircraft Wing-Box. This selection was somewhat arbitrary, but again is consistent with the following criteria. First, both are complex problems: the Wing-Box is an engineering design problem based on real data and constraints, and the Multiple Knapsack is a highly constrained combinatorial optimisation problem known to be *NP*-hard. Second, both problems were available and relatively easy to implement, and third, both have a natural bit string encoding which was a requirement for the study of error thresholds. Additionally, the Wing-Box problem, as originally designed, has a redundant encoding which allowed the study of neutrality in a real-world domain. A non-redundant encoding of this problem was also proposed and tested with the purpose of comparing results with both type of encodings.

It is worth observing, however, that other real-world problems may have very different characteristics from these two problems selected here.

3.2.1 Abstract Fitness Landscapes

Royal Staircase Functions

The *Royal Staircase* [4] class of functions was proposed by van Nimwegen and Crutchfield (1998) for analysing epochal evolutionary search. They justify their particular choice of fitness function both in terms of biological motivations and artificial evolution issues. Although simple, Royal Staircase functions capture some essential elements found on complex problems, namely, highly degenerate genotype-to-phenotype maps, and the existence of extended neutral networks (defined in Section 3.1.3). The working hypothesis is that many real search problems have genotype search spaces which decompose into a number of such neutral networks. This idea is supported by observations in problem domains as diverse as molecular folding (Schuster, 1994), evolvable hardware (Harvey & Thompson, 1996; Vassilev, Miller, & Fogarty, 1999), and evolutionary robotics (Harvey et al., 1997). One symptom of evolutionary search in the presence of neutral networks is the existence of long periods of fitness stasis (search along a neutral network) punctuated by occasional fitness leaps (transitions to a higher neutral network). The Royal Staircase class of fitness functions capture these essential elements in a simplified form (van Nimwegen & Crutchfield, 1998). A formal definition of the Royal Staircase class of functions is given below.

1. Genotypes are specified by binary strings $s = s_1 s_2 \ldots s_L$, $s_i \in \{0,1\}$, of length $L = NK$, where N is the number of blocks and K the number of bits per block.
2. Starting from the first position, the number $I(s)$ of consecutive 1s in a string is counted.

[4]These functions are related to the more familiar *Royal Road* functions (Mitchell et al., 1992).

3. The fitness $f(s)$ of string s with $I(s)$ consecutive ones, followed by a zero, is $f(s) = 1 + \lfloor I(s)/K \rfloor$. The fitness is thus an integer between 1 and $N+1$, corresponding to 1 plus the number of consecutive fully-set blocks starting from the left.

4. The single global optimum is $s = 1^L$; namely, the string of all 1s.

Fixing N and K determines a particular problem or fitness landscape. As an example, Table 3.1 provides an exhaustive listing of a simple Royal Staircase function with N = 2, K = 2.

String	*Fitness*
11 11	3
11 00	2
11 01	2
11 10	2
01 00	1
01 01	1
01 10	1
01 11	1
10 00	1
10 01	1
10 10	1
10 11	1
00 00	1
00 01	1
00 10	1
00 11	1

Table 3.1: Exhaustive listing of a simple Royal Staircase function with number of blocks N = 2, and block size K = 2. The single optimum is the string of all 1s.

NK Landscapes

The *NK* family of landscapes was introduced by Kauffman (1989) in order to have a problem-independent model for constructing fitness landscapes that can gradually be tuned from smooth to rugged. In the *NK* model, N refers to the number of genes in the genotype (i.e. the string length) and K, to the number of genes that influence a particular gene [5]. In other words, the fitness contribution of each gene is determined by the gene itself plus K other genes in the genotype. According to Kauffman, most properties of this model are independent of the alphabet size A, hence the simplest case of $A = 2$ (i.e. bit strings) is here considered.

The fitness of a bit string s of length N is determined as follows. Every bit i contributes to the total fitness of the string. The fitness contribution (f_i) of each bit depends on its value (0 or 1), and on the value of K other bits in the string ($0 \leq K \leq N - 1$). These dependencies are called *epistatic interactions*. Hence, the fitness contribution of one bit depends on the value of $K+1$ bits, giving rise to 2^{K+1} possibilities, called neighbourhood configurations. Each of these

[5]Notice that the meaning of parameters N and K differs from their meaning on the Royal Staircase class of functions.

configurations is assigned a random number uniformly distributed over $[0.0, 1.0]$. Therefore, the fitness contribution f_i of each bit i ($0 \leq i \leq N$) is specified by a lookup table of 2^{K+1} random numbers between 0.0 and 1.0. To compute the fitness of the entire string s, the fitness contribution from each bit is averaged as follows:

$$f(s) = \frac{1}{N}\sum_{i=1}^{N} f_i \tag{3.1}$$

For a given bit i, the set of K epistatic interactions may be either randomly selected or consist of the immediately adjacent bits. Here, the second model of interactions (nearest neighbour) is considered.

Figure 3.2 and Table 3.2 give an example of a NK landscape instance with $N = 5$ and $K = 2$. Bit interactions follow the nearest neighbour model, where the genotype forms a torus. Therefore, the first bit is linked to the last and second bits; the second bit is linked to the first and third; and so on (Figure 3.2). Table 3.2 (the lookup table) shows the fitness contribution of each bit as determined by its value and the value of the two bits to which it has interactions.

Neighbourhood	Bit_1	Bit_2	Bit_3	Bit_4	Bit_5
000	0.968	0.067	0.345	0.653	0.854
001	0.267	0.576	0.021	0.275	0.073
010	0.288	0.174	0.511	0.793	0.139
011	0.915	0.986	0.912	0.287	0.913
100	0.302	0.457	0.521	0.245	0.456
101	0.128	0.233	0.604	0.754	0.543
110	0.698	0.645	0.400	0.237	0.141
111	0.936	0.112	0.313	0.432	0.834

Table 3.2: Lookup table of NK landscape instance ($N = 5, K = 2$).

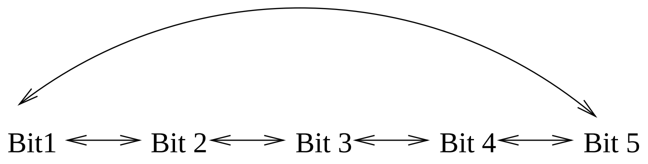


Figure 3.2: Nearest neighbour epistatic interactions. NK landscape ($N = 5, K = 2$).

As an example of how to compute the fitness of a genotype using lookup Table 3.2, let us consider the string ‘00010’. Table 3.3 shows both the neighbourhood configuration and fitness contribution of each bit in ‘00010’ (following lookup Table 3.2). The fitness of the entire string is then computed using Equation 3.1 producing 0.461.

By increasing the value of K from 0 to $N - 1$, NK landscapes can be tuned from smooth to rugged. When K is small, neighbouring strings will have small differences in fitness, because the bits that are different in the two strings will influence the contribution of only few bits in

Bit Position	*Value*	*Neighbourhood*	*Contribution*
1	0	000	0.968
2	0	000	0.067
3	0	001	0.021
4	1	010	0.793
5	0	100	0.456

Table 3.3: Calculating the fitness of ‘00010’. Neighbourhood configuration and fitness contribution of each bit according to lookup Table 3.2.

each string. The extreme case of $K = 0$ yields a single-peaked and smooth ‘Fujiyama’ landscape. When K is large, on the other hand, neighbouring strings will have large differences in fitness, because the differing bits of the two strings will influence the fitness of a large number of bits in each string. When K assumes its largest possible value ($K = N - 1$), the fitness landscape will be completely random or “uncorrelated”, because changing the value of only one bit changes the fitness contribution of all bits in the string, so the overall fitness of neighbouring strings will be totally different. The NK landscape, however, was invented not to explore the two extreme landscapes, but to have a model which allows the construction of an ordered family of tunable correlated landscapes.

NK Landscapes with Neutrality (NKF Landscapes)

Newman and Engelhardt (1997) introduced a family of landscapes with a tunable degree of neutrality (let us call it the NKF model). A similar tunable model with neutrality (NKp model) was also proposed by Barnett (1997). The NKF model is a generalisation of Kauffman’s NK landscape described above. Every bit i makes a contribution f_i to the fitness of the string s. The magnitude of this contribution depends on its value and the value of K neighbouring bits. For a binary alphabet, there are 2^{K+1} possible neighbourhood configurations, and hence 2^{K+1} possible values of f_i. As in Kauffman’s NK model, these values are selected at random. However, in the NK model, the values are random real numbers in the interval $0.0 \leq f_i < 1.0$, whereas in the NKF model the values are *integers* in the range $0 < f_i < F$. For example, if $F = 2$, each contribution f_i is either zero or one. To compute the fitness of the entire string s, the fitness contribution from each bit is averaged as follows:

$$f(s) = \frac{1}{N(F-1)} \sum_{i=1}^{N} f_i \tag{3.2}$$

The fitness of all strings thus falls in the interval [0,1], and there are $NF - N + 1$ possible values in this range. In the limit where $F \to \infty$, the probability that two sequences have the same fitness becomes vanishing small, so the model has no neutrality and is equivalent to the NK model. However, when F is finite, the probability of two sequences having the same fitness is finite. So the model has neutrality, and the degree of neutrality increases as F decreases. Neutrality is greatest when F takes the smallest possible value of 2.

3.2.2 Real-World Domains

Two real-world domains were selected as test problems. They are described below.

Multiple Knapsack Problem

The combinatorial optimisation problem described here, called the 1/0 multiple knapsack problem, follows the specifications given by Khuri et al. (1994). This problem is a generalisation of the 0/1 simple Knapsack problem where a single knapsack of capacity C, and n objects are given. Each object has a weight w_i and a profit p_i. The objective is to fill the knapsack with objects producing the maximum profit P. In other words, to find a vector $x = (x_1, x_2, \ldots, x_n)$ where $x_i \in \{0, 1\}$, such that $\sum_{i=1}^{n} w_i x_i \leq C$ and for which $P(x) = \sum_{i=1}^{n} p_i x_i$ is maximised.

The multiple version consists of m knapsacks of capacities $c_1, c_2, \ldots, c_m$ and n objects with profits $p_1, p_2, \ldots, p_n$. Each object has m possible weights: object i weighs w_{ij} when considered for inclusion in knapsack j ($1 \leq j \leq m$). Again, the objective is to find a vector $x = (x_1, x_2, \ldots, x_n)$ that guarantees that no knapsack is overfilled: $\sum_{i=1}^{n} w_{ij} x_i \leq c_j$ for $j = 1, 2, \ldots, m$; and that yields maximum profit $P(x) = \sum_{i=1}^{n} p_i x_i$. A formal definition of the 1/0 multiple knapsack problem is given below:

Problem instance:

Knapsacks: $1, 2, \ldots, m$

Capacities: $c_1, c_2, \ldots, c_m$

Objects: $1, 2, \ldots, n$

Profits: $p_1, p_2, \ldots, p_n$

Weights: $w_{1j}, w_{2j}, \ldots, w_{nj} (1 \leq j \leq m)$

Capacities and profits are positive numbers while weights are nonnegative.

Feasible solution: A vector $x = (x_1, x_2, \ldots, x_n)$ where $x_i \in \{0, 1\}$ such that:

$\sum_{i=1}^{n} w_{ij} x_i \leq c_j$ for $j = 1, 2, \ldots, m$

Objective function: A function $P(x) = \sum_{i=1}^{n} p_i x_i$, where $x = (x_1, x_2, \ldots, x_n)$ is a feasible vector.

Optimal Solution: A feasible vector that gives the maximal profit; i.e. that maximises the objective function.

This problem leads naturally to a binary encoding. Each string $x_1 x_2 \ldots x_n$ represents a potential solution. If the ith position has the value 1 (i.e. $x_i = 1$) then the ith object is in all knapsacks; otherwise, it is not. Notice that a string may represent an infeasible solution. A vector $x = (x_1, x_2, \ldots, x_n)$ that overfills at least one of the knapsacks; i.e., for which $\sum_{i=1}^{n} w_{ij} x_i > c_j$ for some $1 \leq j \leq m$, is an infeasible string. Rather than discarding infeasible strings and thus ignore infeasible regions of the search space, the approach suggested by Khuri et al. (1994) is to allow infeasible strings to join the population. A penalty term reduces the fitness of infeasible strings. The farther away from feasibility, the higher the penalty term of a string. Thus, the following fitness function was defined (s is the number of overfilled knapsacks):

$$f(x) = \sum_{i=1}^{n} p_i x_i - s \times max(p_i) \tag{3.3}$$

Hence, the fitness function uses a graded penalty term $max(p_i)$. The number of times this term is subtracted from the fitness of a infeasible solution is equal to the number of overfilled knapsacks that solution produces.

Wing-Box Problem

The Wing-Box problem was formulated as part of the Genetic Algorithms in Manufacturing Engineering (GAME) project at COGS, University of Sussex[6]. An industrial partner, British Aerospace, provided data from a real Airbus wing-box. A common problem faced in the design of aircraft structures is to define structures of minimum weight that can withstand a given load. Figure 3.3 sketches the elements of a wing relevant to this problem. The wing is supported at regular intervals by slid ribs, which run parallel to the aircraft's fuselage. On the upper part of the wing, thin metal panels cover the gap separating adjacent ribs. The objective is to find the number of panels and the thickness of each of these panels while minimising the mass of the wing and ensuring that none of the panels buckle under maximum operational stresses. More details, and the equations for calculating the fitness function, can be found in McIlhagga et al. (1996).

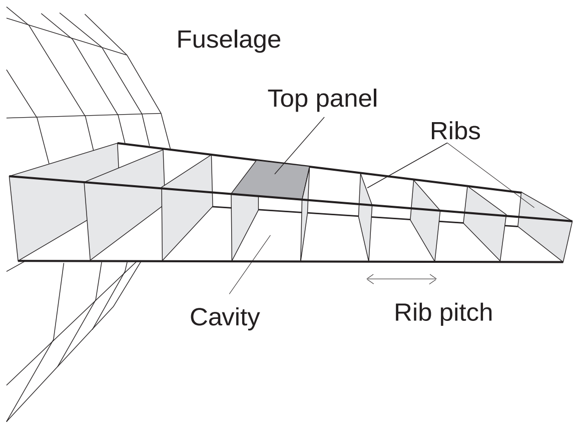


Figure 3.3: Relevant elements of a wing for the Wing-Box problem. Wing dimensions are fixed. The variable elements are the number of ribs and the thickness of top panels.

A full description of a potential solution to the Wing-Box problem requires the definition of the number of panels N and the thickness of each panel. A constraint of the problem is that adjacent panels should not differ in thickness by more than 0.25 mm. The simplest way to accomplish this is to encode the differences in thickness between adjacent panels rather than the absolute thickness of the panels. This design decision, and the Delta encoding described bellow, were proposed by McIlhagga et al. (1996) in the original definition of the problem. Figure 3.4 illustrates the problem representation. Delta of ith Panel denotes the amount by which the thickness of the ith panel is bigger or smaller than the $(i-1)th$ panel.

[6]http://www.cogs.susx.ac.uk/projects/game/

Thickness of 1st Panel	Delta of 2nd Panel	Delta of 3rd Panel	. . .	Delta of Nth Panel

Figure 3.4: Delta encoding. Genetic representation of the wing parameters. For the first panel, the absolute thickness is encoded, whereas for the other panels the differences in thickness between adjacent panels (Deltas) are encoded.

According to the problem specifications, the thickness of the first panel can vary between 8 and 15 mm by steps of 10^{-3} mm. This requires $7 \times 10^3 = 7,000$ values which can be represented with a minimum of 13 bits. But 13 bits can encode 8,192 possible values, so some thickness are represented by more than one binary sequence. This introduces an amount of redundancy in the genotype to phenotype mapping. For all subsequent $N-1$ panels, the difference in thickness relative to the previous panel is encoded. According to manufacturing tolerance considerations, only five values are allowed for these differences in thickness : $\{-0.25, -0.125, 0.0, 0.125, 0.25\}$ (measured in mm). Three bits are needed to encode these five values with the following mapping:

Bit String	Delta (in mm)
000	0.25
001	0.125
010	0.0
011	-0.125
100	-0.25
101	0.25
110	0.125
111	0.0

Table 3.4: Redundant mapping of differences in thickness among consecutive panels in the Wing-Box problem.

Notice that this mapping is redundant since two different triplets represent values 0.25, 0.125, and 0.0. An alternative non-redundant mapping is also proposed here with the purpose of comparing results with both type of encodings. The number of possible differences in fitness is increased from five to eight. The values -0.25 and 0.25 are maintained as the lower and upper bounds, so the same space of solutions is explored, but the positive half of the range is split equally between three values while the negative is split between four (Table 3.5).

For both encodings, the total number of bits needed to represent an individual is $13 + 3 \times N - 1$, that is, 13 for the first panel, and 3 for each of the other N-1 panels.

3.3 Summary

This chapter introduced the notion of fitness landscapes, which was originally proposed in the context of organic evolution, but later gained relevance in both molecular evolution and evolutionary computation. Hence, evolution (natural or artificial) can be seen as an adaptive-walk over a

Bit String	Delta (in mm)
000	0.25
001	0.166
010	0.0833
011	0.0
100	-0.0625
101	-0.125
110	-0.1875
111	-0.25

Table 3.5: Non-Redundant mapping of differences in thickness among consecutive panels in the Wing-Box problem.

fitness landscape. Landscapes may differ in their structure, hence, Section 3.1.3 discussed some landscape features that are known to have an influence on evolutionary search. Among such features **ruggedness** and **neutrality** were distinguished. Thereafter, some techniques for analysing the structure of fitness landscapes were briefly discussed (Section 3.1.4).

The second part of the chapter (Section 3.2) discussed the test problems used throughout this dissertation. Two types of test problems were selected; first, a group of abstract fitness landscapes: Royal Staircase functions, NK landscapes, and NK landscapes with neutrality (Section 3.2.1). These families of tunable landscapes allows the exploration of a wide range of landscape structures with several degrees of ruggedness and neutrality. Second, in order to explore the practical relevance of the ideas in this thesis, two real-world applications were selected (Section 3.2.2), namely, a combinatorial optimisation problem — the Multiple Knapsack problem, and an engineering application — the design of an optimal aircraft Wing-Box. It is worth noticing, however, that other real-world problems might have very different characteristics from these two problems selected here.

Chapter 4

Error Thresholds in Genetic Algorithms: Simple Landscapes

This chapter explores the notions of quasispecies and error thresholds from molecular evolution (Section 4.1.1). The Quasispecies model is based on differential equations and describes the dynamics of replicating nucleic acid molecules under the influence of mutation and selection (Eigen, 1971; Eigen & Schuster, 1979). As noted in the introduction, the *error threshold* of replication is an important notion in this model. The error threshold is a critical mutation rate (error rate) beyond which structures obtained by an evolutionary process are destroyed more frequently than selection can reproduce them. With mutation rates above this critical value, an optimal sequence would not be stable in the population.

The quasispecies model, as stated originally, considered infinite asexual populations. More recently, Boerlijst et al. (1996) extended the quasispecies model by including recombination. Section 4.1.3 describes this work in detail. However, results obtained using infinite population models, cannot be expected to automatically apply to the more realistic case of finite populations. Thus, Section 4.2 reproduces the experiments by Boerlijst et al. (1996) but using a GA — and hence finite populations — instead of the quasispecies model as the underlying model of evolution. An additional group of experiments (not presented in Boerlijst et al. (1996)) are reported in Section 4.3.4. These experiments incorporate mate selection and highlight an advantage of the GA-based simulation model over the analytical model, where eliminating the random-mating assumption would be very difficult (if possible at all).

To summarise, studies on error thresholds from the literature in molecular evolution, are based on the quasispecies analytical model. Hence, the main purpose of the experiments in this chapter is to explore whether a phenomenon similar to that of error thresholds is observed on evolving populations of bit strings using a GA (instead of the quasispecies model) as the underlying model of evolution. This exploration starts by considering two simple abstract landscapes. It is argued in this thesis that the notion of error thresholds is of significance for GAs because it determines a critical upper bound for the balance between exploration and exploitation in genetic search.

4.1 Framework

4.1.1 Quasispecies and Error Thresholds

The *quasispecies* model was introduced by Eigen (1971), Eigen and Schuster (1979) in the context of their work on the origin of life. This analytical model, based on a set of differential equations, describes the cluster of closely related molecular 'species' produced by errors in the self-replication of macromolecules (nucleic acids).

Given an infinite population on a sequence space (described in Chapter 3), and a specified mutation rate governing errors in asexual replication, one can determine the stationary distribution of sequences reached after any transients from some original distribution have died away (Eigen et al., 1988). Unless the mutation rate is too large or differences in fitnesses too small, the population will typically cluster around the fittest sequence(s), forming a concentrated cloud; the average Hamming distance between two members of such a distribution drawn at random will be relatively small. This clustered distribution is called a *quasispecies*.

An important concept in quasispecies theory is the notion of *error threshold* of replication. If replication were error free (i.e. mutation free), no mutants would arise and evolution would stop. On the other hand, evolution would also be impossible if the error rate of replication were too high (only few mutations may produce an improvement, but most will lead to deterioration). The notion of error threshold allows us to quantify the resulting minimal replication accuracy (i.e. maximal mutation rate) that still maintains adaptation (Nowak & Schuster, 1992).

Let us consider an extreme fitness landscape of sequence length L, which contains a single peak of fitness $\sigma > 1$, all other sequences having a fitness of 1. With an infinite population, there is a phase transition at a particular error rate p (the error threshold). This critical error rate was analytically determined by Eigen and Schuster (1979) and it is defined as the rate above which the proportion of the infinite population on the peak drops to chance levels. Following Eigen and Schuster, let $q = 1 - p$ be the per-locus replication accuracy. Then, at the phase transition, the probability of accurate replication of the "master sequence" on the peak needs to be balanced by its superior replication rate, so as to equate with the replication of all the other sequences (back-mutations from these to the master sequence are ignored). Thus,

$$\sigma q^L = \sigma \left[(1-p)^{\frac{1}{p}} \right]^{Lp} = 1 \tag{4.1}$$

and if p is very small, we can approximate the contents of the square brackets by e^{-1}, which leads to

$$p = \frac{ln(\sigma)}{L} \tag{4.2}$$

For mutation rates lower than this critical value (the error threshold) the proportion of master sequences in the population will build up, giving the quasispecies centred on the peak.

4.1.2 Error Thresholds in Finite Asexual Populations

The quasispecies model, as stated originally, considered infinite asexual populations. Later on, Nowak and Schuster (1989) extended the calculations of the error threshold on a single peak landscape to *finite* populations. Finite populations lose grip on the solitary spike of superior fitness easily because of the added hazard of natural fluctuations in this case. In Ochoa and Harvey (1998), we derived a reformulation of the Nowak and Schuster analytical expression. This reformulation, reproduced below, explicitly approximates the extent of the reduction in the error threshold as we move from infinite to finite populations.

Nowak and Schuster (1989) extended the calculations of error thresholds from *infinite* to *finite* asexually replicating populations. Their main result is presented as follows:

> The error threshold can be expanded in a power series of the reciprocal square root of the population size, and this increases with $1/\sqrt{N}$ in sufficiently large populations.

More precisely, the reciprocal square root factor applies to the *difference* between the critical *replication accuracy* per-locus in an infinite population $q_{min}(\infty)$, and the equivalent q_N in a population size N. The reference is to the second term in the following expansion, on the assumption that the third and subsequent terms are relatively insignificant and can be ignored (Nowak & Schuster, 1989).

$$q_N = q_\infty \left(1 + \frac{2\sqrt{\sigma - 1}}{L\sqrt{N}} + \frac{2(\sigma - 1)}{LN} + \frac{(\sigma - 1)^{3/2}}{LN^{3/2}} + \dots\right) \tag{4.3}$$

In many practical circumstances the selection strength σ may lie between 1 and 5; this implies, for values of $N \geq 100$ and of $L \geq 10$, that q_N should differ from q_∞ by only of the order of 4% or less. However, error thresholds are usually reckoned in terms of critical error rates $p = 1 - q$ per-locus; and it turns out that the proportional changes in critical values of p are much more significant in finite populations than the proportional changes in q. Nowak and Schuster (1989) introduced equation 4.3. Here, we derive a reformulation of this equation, which makes explicit the reduction in the critical mutation rate as we move from an infinite to a finite population of size N. In other words, instead of calculating the critical *replication accuracy* (q_N), we calculate the critical *error rate* (p_N):

$$\frac{p_\infty - p_N}{p_\infty} = \frac{2\sqrt{\sigma - 1}(1 - p_\infty)}{L\sqrt{N}p_\infty} \tag{4.4}$$

ignoring further terms in the expansion. Using Equation 4.2 to substitute for p_∞, we have the proportional reduction in the error threshold:

$$\frac{p_\infty - p_N}{p_\infty} = \frac{2\sqrt{\sigma - 1}}{\sqrt{N}} \left(\frac{1}{ln(\sigma)} - \frac{1}{L}\right) \tag{4.5}$$

For large values of L the second term in the bracket is relatively insignificant and can be eliminated producing:

$$\frac{p_\infty - p_N}{p_\infty} \simeq \frac{2\sqrt{\sigma-1}}{ln(\sigma)\sqrt{N}} \tag{4.6}$$

Alternatively, equation (4.5), can be presented as:

$$p_N = \frac{ln(\sigma)}{L} - \frac{2\sqrt{\sigma-1}}{L\sqrt{N}} + \frac{2ln(\sigma)\sqrt{\sigma-1}}{L^2\sqrt{N}} \tag{4.7}$$

Thus, the error threshold for finite populations decreases with the size of the population, given that the second term is subtracting and is the greatest of the series (except the first term).

4.1.3 Viral Quasispecies and Recombination

Most mathematical models describing quasispecies focus on point mutations as the principal source of variation. However, Boerlijst et al. (1996) proposed a mathematical model of quasispecies which incorporates both mutation and recombination. In particular, they study virus populations, which can be modelled as quasispecies. Viruses are infectious agents found in all life forms (plants, animals, fungi and bacteria). A virus particle consist of a core of nucleic acid, which may be DNA or RNA, surrounded by a protein coat. Certain viruses named "retro-viruses" (e.g. HIV) can recombine their genetic material. They carry two copies of their genome in every virus particle, thus, recombination may occur when two distinct strains of the same virus simultaneously infect a single cell. The model of Boerlijst et al. specifically deals with retrovirus recombination. They first considered viral quasispecies without recombination. In their model, distinct viral strains were represented by bit strings of length L. A set of differential equations (see box below) described the change in uninfected cells x, infected cells y_i and free viruses v_i.

$$\frac{dx}{dt} = \lambda - \delta x - x\sum_i \beta_i v_i \tag{4.1}$$

$$\frac{dy_i}{dt} = x\sum_j Q_{ij}\beta_j v_j - a_i y_i \tag{4.2}$$

$$\frac{dv_i}{dt} = k_i y_i - u_i v_i \tag{4.3}$$

In this model λ is the influx rate of uninfected cells; δ, a_i and u_i are the death rates of, respectively, uninfected cells, infected cells, and free virus; β_i is the infection rate; k_i the production rate of new free virus; and Q_{ij} is the probability of strain j mutating to strain i. The mutation matrix is given by:

$$Q_{ij} = p^{H_{ij}}(1-p)^{L-H_{ij}} \tag{4.4}$$

Here p is the mutation rate per bit, L is the bit string length, and H_{ij} is the Hamming distance between strings i and j. Error free replication is given by $Q_{ij} = (1-p)^L$.

The next paragraph describes the results obtained with this model without recombination. Following the notation of Boerlijst et al., p stands for the mutation rate per bit, whereas p_c indicates the critical mutation rate (or error threshold).

Without mutation ($p = 0$), the strain (or sequence) with the largest *reproductive ratio* (or fitness) R_i will out-compete all other sequences. However, with mutation ($p > 0$) there is a critical error rate, p_c, beyond which the sequence with the highest R_i is no longer preferentially selected. A single peak fitness landscape is considered where a sequence, F, has the highest fitness R_F, and all other sequences have the same, but lower, fitness R. If $p < p_c$, the quasispecies will be centred around the fittest sequence, F, which will be the most abundant. However, if $p > p_c$, the fittest sequence, F, will not be preferentially selected and each virus sequence will have essentially the same relative abundance. This phenomenon is known as the error threshold (Section 4.1.1).

A Basic Principle of Recombination

Before including recombination in the model, Boerlijst et al. discussed a relation which holds for **any** type of recombination, and turns out to be an important element for understanding the effect of recombination on error thresholds.

> Consider two sequences i and j with a genetic distance d_{ij} (for a bit string model d_{ij} is the Hamming distance). Assume that these sequences recombine to produce an offspring k. If recombination is the only source of variation, we have
>
> $d_{ik} + d_{jk} = d_{ij}$.
>
> The genetic difference between the parents equals the sum of the genetic difference between offspring and each of the parents. This relation is important for our understanding of recombination. It shows that in sequence space recombination is always inwards pointing [(Boerlijst et al., 1996), p. 1578].

To illustrate this principle, consider the following example:

i: 1001011101, j: 0000000000, d_{ij} = 6

k: 0001001101, is one possible product of uniform recombination between i and j.

d_{ik} = 2, d_{jk} = 4, $d_{ik} + d_{jk}$ = 4 + 2 = 6 = d_{ij}

Bit string recombination model

Boerlijst et al. (1996) extended the mathematical model by including recombination. They introduced variables for double infected cells and for viruses produced by these cells. Double infected cells y_{ij} are those infected by two strains i and j. v_{ij} represents the free virus produced by these double infected cells, of which 25% will be homozygous type i, 25% will be homozygous type j, and 50% will be heterozygous (i.e. will have one i and one j strain). Due to this characteristic of the model, the recombination rate r has a maximum at $r = 0.5$, because only heterozygous virus particles can (effectively) recombine. To model recombination 'uniform crossover' (Syswerda, 1989) was used. The new set of equations is shown in the box that follows.

$$\frac{dx}{dt} = \lambda - \delta x - xV \tag{4.5}$$
$$\frac{dy_i}{dt} = xV_i - a_i y_i - sy_i V \tag{4.6}$$
$$\frac{dy_{ij}}{dt} = sy_i V_j - a_{ij} y_{ij} \tag{4.7}$$
$$\frac{dv_i}{dt} = k_i y_i - u_i v_i \tag{4.8}$$
$$\frac{dv_{ij}}{dt} = k_{ij} y_{ij} - u_{ij} v_{ij} \tag{4.9}$$

Here s is the rate of double-infection, $V = \sum_i \beta_i v_i + \sum_{ij} \beta_{ij} v_{ij}$ is the sum of all infectious virus and $V_i = \sum_j Q_{ij} \beta_j v_j + \sum_j Q_{ij} \sum_{kl} M_{jkl} \beta_{kl} v_{kl}$ is the sum of infectious virus of type i, after mutation and recombination, with M_{jkl} being the probability of strain k and l recombining to strain j. All other variables and parameters are as described in equations (4.1)–(4.3).

The authors studied the steady state structure of the population using the new model including recombination. The following simulation parameters were used: string length of 15 and recombination rate of $r = 0.5$. Two abstract fitness landscapes, (a) Single peak and (b) Plateau, were considered.

(a) Single peak landscape First, the case where only one sequence F has an increased fitness, was studied. This single bit string has fitness $R_F = 5$, whereas all other sequences have fitness $R_i = 3.5$. The steady state distribution of sequences for this landscape shows that, for an error rate of $p = 0.007$, the population with recombination (compared against the population without recombination) is more compact in that there are less distant sequences; but there is also less of sequence F. This effect of recombination can be understood as follows. Most of the population is of sequence F. If sequence F recombines with e.g. a sequence in Hamming distance class 8, then the offspring lies anywhere between F and H_8 (according to the basic principle of recombination discussed above). However, for a slightly increased error rate of $p = 0.008$, recombination drives the population beyond the error threshold, resulting in an almost uniform distribution of sequences. This behaviour is qualitatively similar to that obtained empirically using a GA for finite populations (Section 4.2), thus Figure 4.1 illustrates similar distributions, although for different mutation values. Thus, where recombination acts as a converging operator when F is involved, it acts as a diverging operator in other cases. If, for instance, two sequence in H_4 recombine, the product lies anywhere between F and H_8. Recombination introduces instability to the population composition; it shifts the error threshold, but at low mutation rates, it can make the population more compact.

(b) Plateau landscape In the single peak landscape, recombination is disadvantageous for the virus, because it decreases the abundance of F and introduces instability to the population (shifts the error threshold towards lower mutation rate). Recombination, however, can be advantageous in the case of correlated landscape. Consider a situation, where the fitness of sequences close by the fittest string F is increased to $R_{H_1} = 4.8$, and $R_{H_2} = 4.6$. Where H_1 is the set of all sequences with a Hamming distance of 1 from the fittest string F, and H_2 the set of all sequences with a Hamming distance of 2 from F. In this case, the steady state distribution of sequences shows that, before the error threshold at $p = 0.011$, the population with recombination is again more compact, and it has more of its ‘mass’ in the middle of the fitness plateau. However, if the error rate is increased, at a certain point (around $p = 0.015$), and fairly suddenly, recombination can no longer keep the population in the middle of the fitness plateau (Figure 4.2 mirrors these results, although for different mutation values). Notice that, in this scenario, recombination increases the abundance of the fittest sequence F (Figure 4.2 (a)). Thus, recombination may be advantageous to the virus but only for small mutation rates (below the error threshold).

Boerlijst et al. main conclusions may be summarised as follows:

- For small mutation rates (i.e. below the error threshold), recombination can focus the quasispecies around a fitness optimum.
- Recombination shifts the error threshold to lower mutation rates, and makes the transition sharper.
- Recombination is advantageous (in the sense that it increases both the proportion of the fittest string in the population, and the average fitness of whole population) if fitness is more correlated (as in the plateau landscape) and if the mutation rate is sufficiently small.

Finally, the authors report (although not showing the results) that they have extensively tested the bit string model for other fitness distributions such as ‘smooth’ fitness peaks, multiple peaks and random distributions; looked into alternatives to uniform crossover, such as one-point and multi-point crossover; and that in all these cases, the main conclusion holds — recombination shifts the error threshold towards lower mutation rates and makes the transition sharper.

4.2 Experiments

The previous section described results obtained with the analytical quasispecies model, including both mutation and recombination for infinite populations. In contrast, the experiments described in this section use a GA and hence finite populations, as the underlying model of evolution. These experiments reproduce the study described above, now in the scenario of discrete finite populations.

All the experiments used a generational GA with fitness proportional selection. The genetic operations were uniform crossover and the standard bit mutation. The GA was run in two modes *Asexual*: using mutation only; and *Sexual*: using both mutation and recombination. The landscapes explored were those described in the previous section (the single peak and plateau landscapes). For the purposes of the simulation, the fittest string, F, was always the string of all 0s (000000000000000) with no loss of generality. Any other bit string is referred to as a ‘mutant’, and belongs to one of the Hamming distance classes H_i, where i is the Hamming distance[1] to F. In the simulations, the initial population was generated differently for each landscape. For the single peak landscape, around 50% of the population was set on the peak and the rest was randomly generated. For the plateau landscape, 25% was set on the peak, 25% on the H_1 compartment, 25% on the H_2 compartment, and the rest was randomly generated. Two population sizes were tested (100 and 1000), with the aim of studying the effect of population size on the magnitude of error thresholds. The per bit mutation rate p was varied from $p = 0.005$ to $p = 0.04$, with a step size of 0.005. The number of generations per GA run was 500. This value was empirically selected;

[1]Given that the F is the string of all 0s, the Hamming distance from a given string to F is the number of ones in that bit string.

the distribution of sequences was fairly stable by this point in all cases. Each experiment was run 50 times and the results were averaged. Table 4.1 summarises the GA parameters used in the simulations.

Chromosome length	15
Population size	100, 1000
Recombination rate	0.0 (Asexual), 0.5 (Sexual)
Mutation rate (per bit)	0.005 to 0.04, Step = 0.005
Generations	500
Trials per GA run	50

Table 4.1: GA parameters used in the simulations.

4.3 Results

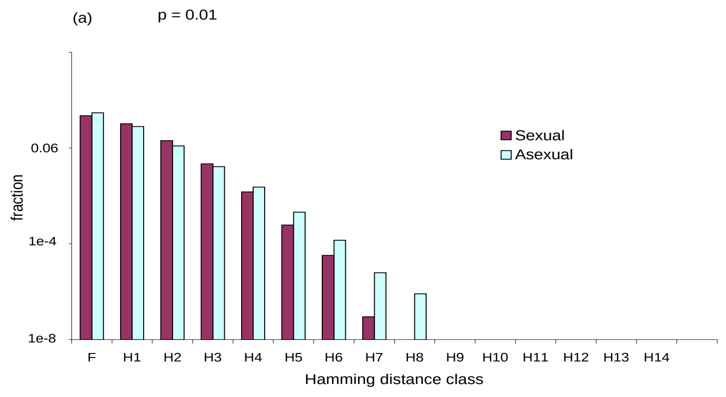
The empirical results using the GA model (for finite populations) mirrored qualitatively those produced by Boerlijst et al. for infinite populations (described in Section 4.1.3). However, error thresholds for finite populations were, in all scenarios, lower than for the infinite case. The following subsections discuss the results obtained with the GA simulation model on both the single peak and plateau landscapes.

4.3.1 Single Peak Landscape

Figure 4.1 show the distribution of sequences, above and below the error threshold of the sexual population, on the single peak landscape for a population size of 1000. These plots, using logarithmic scale for the vertical axis, are almost mirror images of those shown in (Boerlijst et al. (1996), p. 1579) for infinite populations. Figure 4.1(a) shows the distribution of sequences for an error rate of $p = 0.01$, with and without recombination. As for the infinite population (Section 4.1.3), the sexual population is, in some sense, more compact (less diverse); there are fewer distant mutants, but there is also fewer of sequence F. On the other hand, Figure 4.1(b) shows that for a slightly increased error rate ($p = 0.015$) recombination drives the population beyond the error threshold, resulting in an almost uniform distribution of sequences. Although the majority of the sexual population lies in the H_7 and H_8 compartments, it is because these contain the most sequences; each distinct single sequence has approximately the same proportion in the population. In other words, although the distribution of sequences for the sexual population has a curved shape, it is really representing a uniform distribution of sequences. The explanation suggested by Boerlijst and co-workers for this effect of recombination has already been discussed in Section 4.1.3.

4.3.2 Plateau Landscape

Figure 4.2(a) shows the distribution of sequences in the plateau landscape for an error rate $p = 0.02$, using a linear scale. Notice that, for both sexual and asexual reproduction, the majority of the population now lies in the H_2 compartment. Recombination between two H_2 sequences generates offspring anywhere between F and H_4. Recombination thus shifts part of the population back to the middle of the fitness plateau. However, for a slightly increased error rate, $p = 0.025$ (Figure 4.2(b)), recombination drives the population beyond the error threshold.



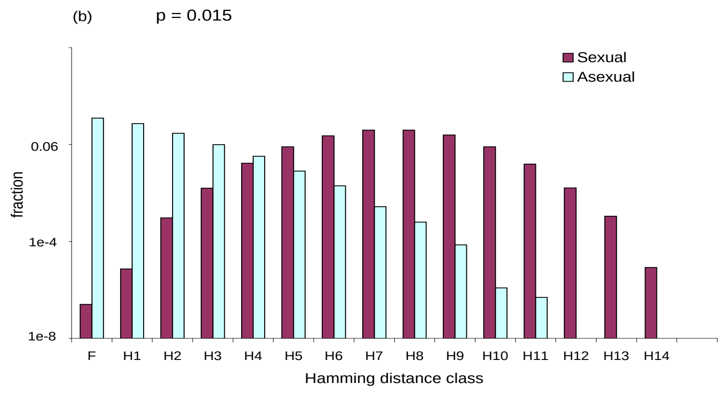
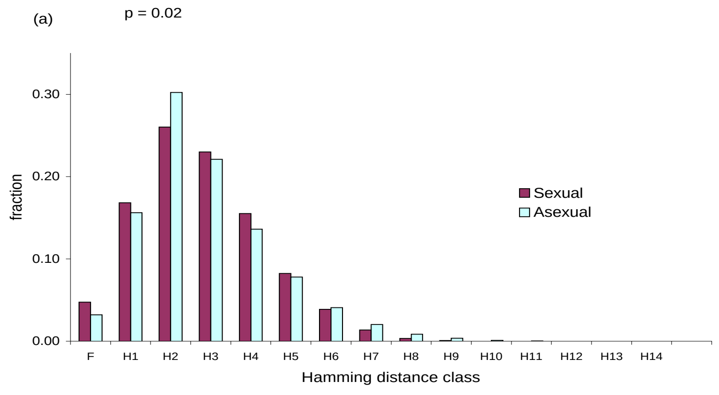


Figure 4.1: Effect of recombination on the error threshold (single peak landscape). Population size = 1000, recombination rate = 0.5. A logarithmic scale is used for the vertical axis. (a) Below the error threshold ($p = 0.01$) the sexual population is more compact. (b) For a slightly increased per bit mutation rate $p = 0.015$, recombination can push the population over the error threshold. H_i denotes the sum of all mutants with a Hamming distance i to F (the fittest string).

4.3.3 Population Size and the Magnitude of the Error Threshold

In contrast to infinite populations, when studying error thresholds for finite populations, it is important to explore the effect of the population size on the magnitude of error thresholds. Thus, experiments in this section explore error thresholds for population sizes of 100 and 1000. Figures 4.3 and 4.4 show error thresholds on both the single peak and plateau landscapes for the two population sizes studied (100 and 1000), respectively.

For infinite populations on a single peak landscape, the definition of the error threshold is straight forward (there is a clear phase transition). However, this is not the case for finite populations where the transition is less sharp. Moreover, if fitness is more correlated, as in the plateau landscape, the transition is even less noticeable. Nevertheless, an error threshold can be identified visually for finite populations, with some degree of uncertainty, as the mutation rate just before the error classes become equally distributed (i.e. the lines become parallel) (Figures 4.3 and 4.4).



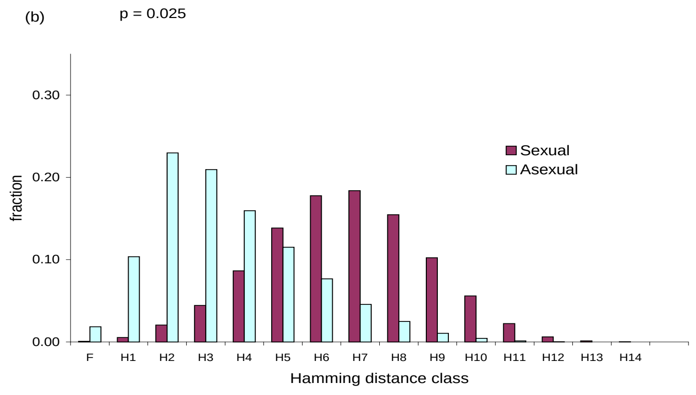


Figure 4.2: Effect of recombination on the error threshold (plateau landscape). Population size = 1000, recombination rate = 0.5. A linear scale is used for the vertical axis. (a) Below the error threshold ($p = 0.02$) the sexual population is again more compact, and it has more of its mass in the middle of the fitness plateau. (b) For a slightly increased mutation rate $p = 0.025$, recombination can no longer keep the population in the middle of the fitness plateau.

	Single Peak			Plateau		
	100	1000	∞	100	1000	∞
Asexual	0.015 (0.0153)	0.020 (0.0211)	(0.0238)	0.030	0.035	**0.05**
Sexual	0.010	0.010	**0.0075**	0.020	0.020	**0.011**

Table 4.2: Error thresholds for finite populations (sizes 100 and 1000) and infinite populations on both the single peak and plateau landscapes. Values were obtained from three sources: (1) empirically using the GA model (shown in normal font), (2) using analytical Equations (shown in brackets), and (3) using the quasispecies analytical model (shown in bold font).

Table 4.2 reports error threshold values for finite populations (sizes 100 and 1000) and infinite populations on both the single peak and plateau landscapes. Values in the table were obtained from three different sources:

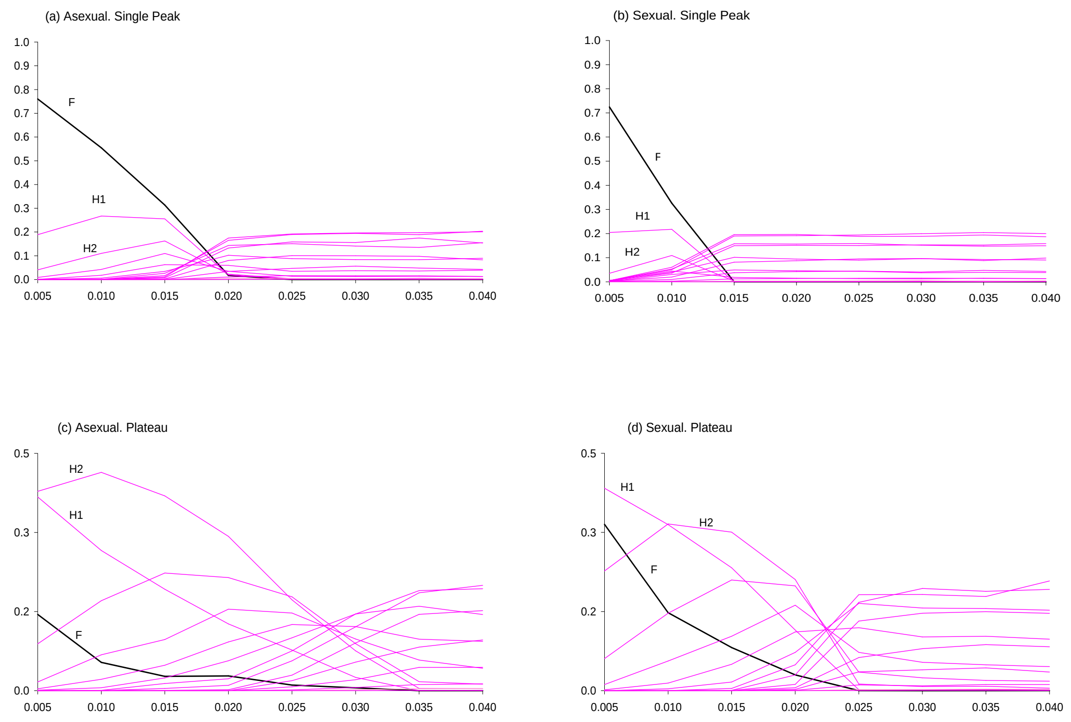


Figure 4.3: Distribution of sequences for a population of size 100. For both asexual and sexual populations in the two abstract landscapes studied. The per bit mutation rate varies from 0.005 to 0.04 with a step of 0.005. An error threshold can be identified visually as the mutation rate just before the error classes become equally distributed (the lines become parallel). Vertical axis shows population fractions, horizontal axis shows per bit mutation rates. The fittest string F and error classes H_1 and H_2 are indicated in the plots.

1. For finite populations, error thresholds were empirically estimated to the nearest step size of 0.005. These values were identified visually from Figures 4.3 and 4.4 as described above.
2. For an infinite asexual population on the single peak landscape, the error threshold was calculated analytically using Equation 4.2. Similarly, for asexual populations of size 100 and 1000 on the single peak landscape, error thresholds were calculated using Equation 4.7. These values are shown in brackets.
3. For infinite sexual populations on both landscapes studied, and infinite asexual populations on the plateau landscape, error threshold values were taken from Boerlijst et al. (1996). The authors obtained these values by running their analytical quasispecies model (described in Section 4.1.3). These values are shown in bold font.

Notice that empirical error thresholds for finite asexual populations (sizes 100 and 1000) on the single peak landscape, have a very good agreement with theoretical values for this landscape calculated using Equation 4.7 (shown in brackets), which validates the empirical approach.

The major trends in Figures 4.3 and 4.4; and Table 4.2, can be summarised as follows:

- Error thresholds for sexual populations are, in all scenarios, lower than for asexual populations.

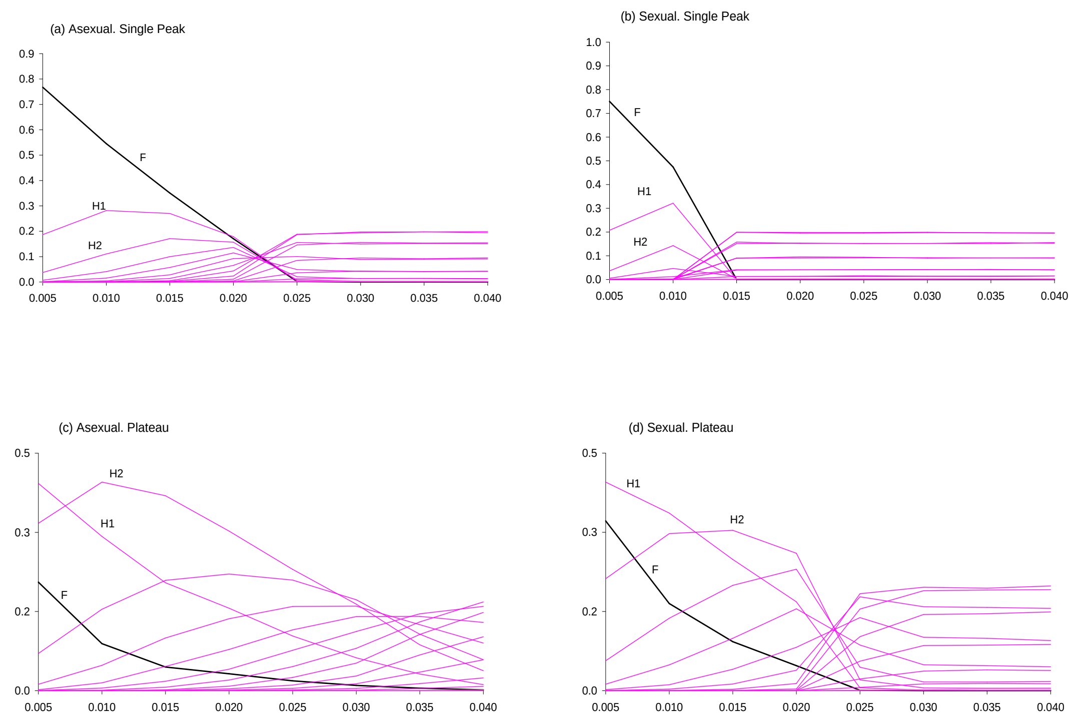


Figure 4.4: Distribution of sequences for a population of size 1000. For both asexual and sexual populations in the two abstract landscapes studied. The per bit mutation rate varies from 0.005 to 0.04 with a step of 0.005. Vertical axis shows population fractions, horizontal axis shows per bit mutation rates.

- For asexual populations, the larger the population the higher the error threshold.
- Error thresholds, in all scenarios, are higher on the more correlated fitness landscape studied (the plateau landscape).
- The transition in the distribution of sequences around the error threshold is sharper for sexual populations as compared to asexual populations.

4.3.4 Mate Selection and the Magnitude of the Error Threshold

In nature, mating is rarely random. Instead, organisms often select their mates following certain criteria. When choice of mates is based on phenotype, mating is called *assortative* (Hart & Clark, 1997). In positive assortative mating (often called simply assortative mating), individuals tend to choose mates that are phenotypically like themselves. In negative assortative mating (also called *dissortative* mating), individuals tend to choose mates that that are phenotypically unlike themselves. Of course, even with random mating, some mating pairs are phenotypically similar or dissimilar, so assortative mating refers only to those situations in which mating partners are phenotypically more similar or dissimilar than would be expected by chance in random mating populations.

The experiments presented in this section, incorporate mate selection into the GA simulation model. Specifically, they include assortative and dissortative mating. These experiments highlight an advantage of the GA model over analytical models, where eliminating the random-mating assumption is normally very difficult. In GAs the phenotype of an organism is summarised by its fitness value. However, the fitness landscapes explored in this chapter are far too simple, having very few different fitness values (two for the single peak landscape, and four for the plateau landscape). Preliminary experiments using assortative mating based on similarity in fitness showed no noticeable differences. However, a second implementation based on Hamming distance between parents, produced notable differences in the error threshold magnitudes among sexual populations with random, assortative and dissortative mating. The experiment parameter settings were similar to those described in Section 4.2, with the difference that the per bit mutation rate covered a wider range (from 0.005 to 0.05 with a step of 0.005). For assortative mating two extra low mutation values (0.0 and 0.001) were also tested. Mate selection was implemented as follows. Survival was still based on fitness, but when choosing a partner for a given individual, two potential mates were selected, and from this pair the closest, according to the mate selection criterion, was taken. Figures 4.5 and 4.6 show the steady state distribution of sequences for a population of size 100 using 4 different reproductive strategies: (a) asexual, (b) sexual with random mating, (c) sexual with assortative mating and (d) sexual with dissortative mating; on the single peak and plateau landscapes respectively.

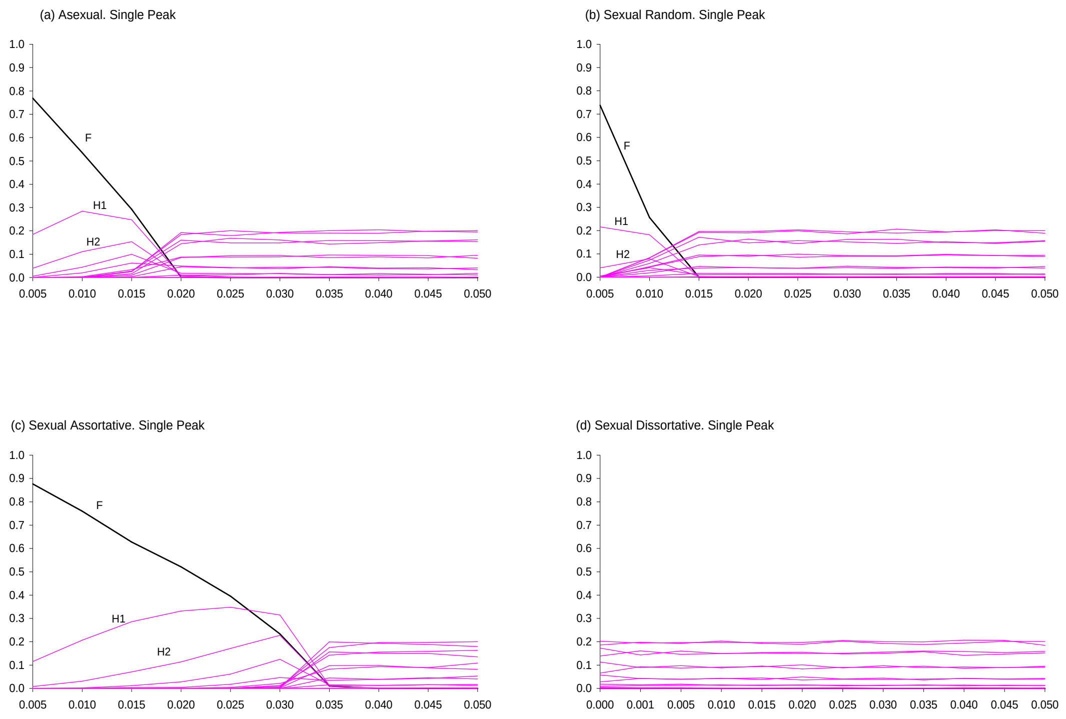


Figure 4.5: Distribution of sequences on the single peak landscape for 4 different reproductive strategies: (a) Asexual, (b) Sexual with random mating, (c) Sexual with assortative mating, and (d) Sexual with dissortative mating. The recombination rate used was r = 0.5. The mutation rate per bit varies from 0.005 to 0.05 with a step of 0.005. For dissortative mating two extra low mutation values (0.0 and 0.001) were tested. Vertical axis shows population fractions, horizontal axis shows mutation rates (per bit).

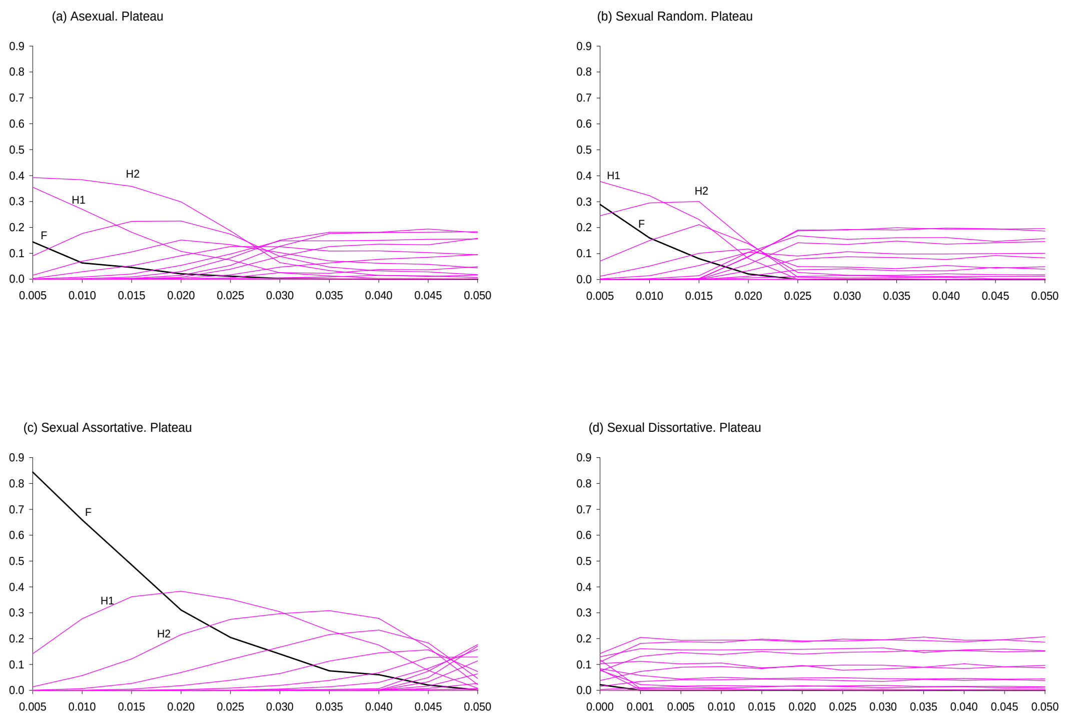


Figure 4.6: Distribution of sequences on the plateau landscape for 4 different reproductive strategies: (a) Asexual, (b) Sexual with random mating, (c) Sexual with assortative mating, and (d) Sexual with dissortative mating. The recombination rate used was r = 0.5. The per bit mutation rate varies from 0.005 to 0.05 with a step of 0.005. For dissortative mating two extra low mutation values (0.0 and 0.001) were tested. Vertical axis shows population fractions, horizontal axis shows mutation rates (per bit).

	Single Peak	*Plateau*
Asexual	0.015	0.030
Sexual Random	0.010	0.020
Sexual Assortative	0.030	0.045
Sexual Dissortative	0.0	0.0

Table 4.3: Error thresholds for a population of size 100 on both the single peak and plateau landscapes, for 4 different reproductive strategies.

Table 4.3 summarises the error thresholds values as identified visually from Figures 4.5 and 4.6, for the 4 reproductive strategies studied. Results suggest that assortative mating, on both fitness landscapes, increases considerably the error threshold on sexual populations as compared to random mating. Error thresholds with assortative mating in sexual populations are even larger than for asexual populations. Moreover, assortative mating seems to be advantageous for the virus population, because it increases the abundance of *F* (see Figures 4.5 and 4.6), and makes the population more stable as the error threshold moves to higher values.

A possible explanation for this effect of assortative mating is as follows. Recombination acts as a converging operator when F is involved, it acts as a diverging operator in other cases. The converging effect of recombination is increased when mating is assortative, because parents tend to be close to each other in Hamming distance, so, when F is involved, offspring will be either F or close to F. This, therefore, increases further the abundance of F and causes an effect such that, in more recombination events F will be involved (even for high mutation rates). Thus, the overall effect of assortative mating is to make the population more compact and to increase the abundance of F, even for mutation rates higher than the error thresholds for a sexual population with random mating.

In contrast, populations with dissortative mating don't show a clear error threshold transition. Even for very low mutation rates, all sequences are equally distributed at the steady-state equilibrium of the population. Dissortative mating seems to be disadvantageous for the virus population, because it decreases the abundance of F even for very low mutation rates; it also introduces instability to the population. A possible explanation for this effect is as follows. Dissortative mating has a diverging effect, because parents tend to be further away from each other. Even when F is involved in a recombination event, the most suited partner, according to the dissortative criterion, will be the farthest away in Hamming distance. If, for instance, a sequence in H_8 is selected as the second parent, the offspring will lie anywhere between F and H_8. Thus, the overall effect of dissortative mating is to decrease the abundance of F and to drive easily (even for very low mutation rates) the recombinant population beyond the error threshold.

4.4 Discussion

For finite populations, and in both abstract fitness landscapes studied, the stable distribution of sequences was qualitatively similar to that for infinite populations. However, error thresholds were smaller in most scenarios for finite populations as compared to infinite populations. Moreover, for asexually replicating populations, the smaller the population, the lower the error threshold. Nevertheless, the several implications of recombination found by Boerlijst and co-workers for viral quasispecies, hold for GAs and finite populations. These implications can be summarised as follows. First, for small mutation rates (i.e. below the error threshold), recombination can focus the population around a fitness optimum. In this sense, recombination acts as an error repair mechanism, but it also means that the population is less flexible to environmental changes. Second, recombination shifts the error threshold to lower mutation rates. Near the error threshold, without recombination, the fittest sequence only makes up a small percentage of the total population (Eigen et al., 1988). Under such conditions, recombination acts as a diverging operator, driving the population beyond the error threshold. Recombination can be advantageous for the population if fitness is correlated and if the mutation rate is sufficiently small.

The relevance of these results to the theory of GAs is twofold. First, in the study of optimal mutation rates, given the relationship between error thresholds and optimal mutation rates postulated in this thesis. Second, in understanding both the role of recombination, and the interaction between recombination and mutation in the behaviour of GAs.

Notice that the study of error thresholds presented in this chapter considered only non-elitist GAs. Elitism will have a critical effect on the error threshold phenomenon. The next chapter will study the effect of elitism on error thresholds for complex landscapes.

Although this chapter studied simple fitness landscapes, the single peak landscape is an extreme case in the continuum of less rugged (or more correlated) landscapes. The plateau landscape is a less extreme case, which also showed distinct behaviours below and above a critical mutation rate. The relationship between error thresholds and optimal mutation rates, postulated in this thesis, will be empirically assessed in Chapter 6. If this relationship turns out to exist, higher values for mutation rates should generally be used in GAs for practical applications. Moreover, the following general suggestions, could be made:

- Given that error thresholds are inversely proportional to genotype length; 'optimal' per bit mutation rates should also hold this relationship to genotype length. This finding has been independently reported several times in the evolutionary computation literature (see Chapter 2).
- Given that error thresholds were shown to be lower for small-sized populations, the per bit mutation rate should be lower the smaller the population size.
- Given that recombination shifts the error threshold to lower mutation rates, the mutation rate per bit should be smaller when recombination is used.
- Given that recombination was shown to increase the population average fitness in more correlated landscapes, the more correlated a fitness landscape is, the more the advantage of using recombination.

These suggestion will be tested in Chapter 6 using more realistic fitness functions. However, the simple abstract landscapes used in this chapter, were useful as a starting point for exploring evolutionary dynamics, and testing hypotheses regarding the roles of genetic operators in GAs.

Finally, a computational 'micro-analytical' (or 'agent-based') simulation model — in this case the GA — can offer some advantages over an analytical model for evolutionary biology studies. In particular, there is the possibility of modifying the general assumption of random mating, allowing instead more biologically inspired patterns of sexual selection. Experiments including mate selection (reported in Section 4.3.4) showed that assortative mating can considerably increase the critical error rate.

4.5 Summary

This chapter introduced the notions of quasispecies and error thresholds from molecular evolution. It also discussed two major extensions of the original quasispecies model within the molecular evolution literature, namely, the analysis of finite asexual populations on a single peak landscape (Nowak & Schuster, 1989) (discussed in Section 4.1.2); and the viral quasispecies model including both mutation and recombination (Boerlijst et al., 1996) (discussed in Section 4.1.3). This latest work explored the effect of recombination on error thresholds for infinite populations on two simple fitness landscapes. The experiments described in Section 4.2, reproduced these results but using a GA (and thus finite populations) instead of the viral quasispecies model (for infinite populations) as the underlying model of evolution. For finite populations and in both abstract fitness landscapes studied, the stable distribution of sequences was qualitatively similar to that for infinite populations. Thus, error thresholds were shown to exist in finite populations of bit strings evolving using a GA. Moreover, the main conclusions of Boerlijst and co-workers, summarised in

Section 4.1.3, hold in this case. However, error thresholds were smaller, in most scenarios, for finite populations. Also, for asexually replicating populations, the smaller the population, the lower the error threshold. On the single peak landscape, the empirical results for asexually replicating populations were accurately predicted by the analytic expression presented in Section 4.1.2. Additional experiments including mate selection were presented in Section 4.3.4. It was found that assortative mating, i.e. preference for similar organisms, increased the magnitude of error thresholds as compared to both asexual replication and sexual replication with random mating. Assortative mating seems to be advantageous for the population as it increases the abundance of the fittest string.

This chapter studied very simple fitness landscapes. However, these landscapes were useful as a starting point to demonstrate the existence of error thresholds in GAs. The next chapter will study the existence of error thresholds in GAs running on more complex and correlated landscapes. It will also explore the effect of both changing various evolutionary parameters and modifying the structure of the fitness landscape, on the magnitude of error thresholds.

It is argued in this thesis that the notion of error thresholds is relevant to GAs applied to complex problems: first, in the study of optimal mutation rates, as error thresholds measure an optimal balance between exploration and exploitation in evolutionary algorithms; second, because knowledge about error thresholds suggests how different evolutionary parameters interact with one another in evolutionary dynamics. This knowledge may have practical implications in suggesting heuristics for effective setting of GA parameters. These issues will be explored further in Chapter 6.

Chapter 5

Error Thresholds in Genetic Algorithms: Complex Landscapes

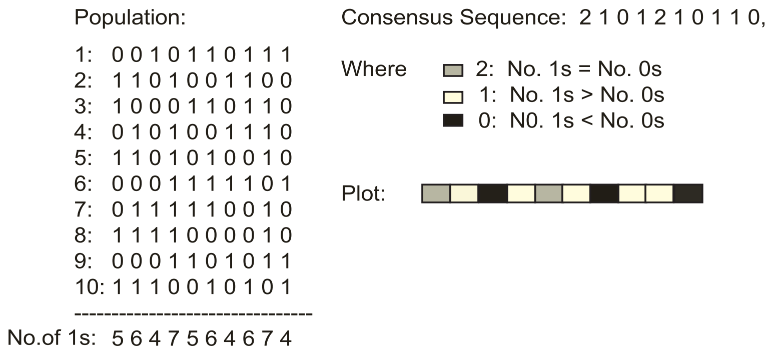
In the previous chapter the existence of error thresholds was demonstrated on two simple landscapes (isolated peak, and plateau) using a standard GA; it was also shown that recombination, in those landscapes, shifts error thresholds toward lower values. This chapter extends those findings by studying more complex landscapes, including real-world domains. The division between simple and complex is somewhat arbitrary. The isolated peak landscape is an extreme uncorrelated landscape, the plateau is less extreme but still highly uncorrelated. This chapter, on the other hand, explores correlated landscapes.

The chapter is organised in four main sections. The method section (Section 5.1) describes the *consensus sequence* plots. These plots, borrowed and adapted from theoretical biology, constitute an empirical approach for locating error thresholds on general landscapes. Section 5.2 uses consensus sequence plots, on two fixed abstract problems, to explore the effect of changing various evolutionary parameters on the magnitude of error thresholds. Thereafter, Section 5.3 uses a fixed GA (fixed evolutionary parameters) and explores the effect of modifying the landscape structure on the magnitude and characteristics of error thresholds. This exploration uses the families of tunable abstract landscapes described in Chapter 3 (Section 3.2). The closing empirical section of the Chapter (Section 5.4), explores whether error thresholds may be identified on real-world applications. It uses the two real-world domains discussed in Chapter 3 (Section 3.3), and explores error thresholds with and without recombination.

5.1 Method

5.1.1 Consensus Sequence Plots

The work of Bonhoeffer and Stadler (1993) studied the evolution of quasispecies on two correlated fitness landscapes, the Sherrington Kirkpatrick spin glass and the Graph Bipartitioning landscape. The authors described an empirical approach for locating thresholds on complex landscapes. In this section, this approach is borrowed and adapted. Instead of the quasispecies model, a GA is used as the underlying model of evolution. The resulting method can be applied for identifying error thresholds in GAs running on general complex landscapes. The approach is to calculate and plot the consensus sequence at equilibrium for a range of mutation rates. The consensus sequence in a population is defined as the sequence of predominant symbols (bits) in each position; it is plotted as follows: if the majority of individuals has a ‘1’ or ‘0’ in a position *i* the field is plotted white or black, respectively. The field is plotted grey if the position is undecided. Figure 5.1, shows an hypothetical population and calculates its consensus sequence. The plot shown in Figure 5.1 will correspond to a single line in a full consensus sequence plot where the per bit mutation rate is varied (see Figure 5.4 for an example of a full plot). The *equilibrium state* is reached when the proportion of different sequences in the population is stationary. This happens when evolution is simulated for a large enough number of generations. In practice, it is considered that the equilibrium is reached when several parameters of the population (e.g. the maximal and average fitness) reach equilibrium. According to Bonhoeffer and Stadler (1993) the error threshold may be approached from *below* or *above*, with both methods producing similar results.



```
Population:                       Consensus Sequence:  2 1 0 1 2 1 0 1 1 0,

1:   0 0 1 0 1 1 0 1 1 1          Where   ▨ 2:  No. 1s = No. 0s
2:   1 1 0 1 0 0 1 1 0 0                  □ 1:  No. 1s > No. 0s
3:   1 0 0 0 1 1 0 1 1 0                  ■ 0:  N0. 1s < No. 0s
4:   0 1 0 1 0 0 1 1 1 0
5:   1 1 0 1 0 1 0 0 1 0
6:   0 0 0 1 1 1 1 1 0 1          Plot:   ▨□■□▨□■□□■
7:   0 1 1 1 1 1 0 0 1 0
8:   1 1 1 1 0 0 0 0 1 0
9:   0 0 0 1 1 0 1 0 1 1
10:  1 1 1 0 0 1 0 1 0 1
     --------------------------------
No.of 1s:  5 6 4 7 5 6 4 6 7 4
```

Figure 5.1: Calculating and plotting the consensus sequence of a population.

Approaching the error threshold from below

To approach the error threshold from below, the simulation starts with a homogeneous population at the global optimum. This approach requires knowing the optimal string beforehand. Then, the population is allowed to reach equilibrium at a constant mutation rate of 0.0. Afterwards, the mutation rate is increased by a fixed, small step and the computation is continued with the current population. This process is repeated until a predefined maximum for the mutation rate is reached. Figure 5.2 outlines this algorithm. Notice that the plot summarises a single run, there is no averaging of multiple runs.

Approaching the error threshold from above

To approach the error threshold from above, the simulation starts with a random population. Then the population is allowed to reach equilibrium at a constant predefined maximum for the mutation rate[1]. Afterwards, the mutation rate is decreased by a fixed small step and the computation continues with the current population. This process is repeated until the mutation rate is 0.0. Figure 5.3 outlines this algorithm. Notice that, in this case, it is not necessary to know the optimal string. Hence, in principle, this approach can be used for locating the error threshold on any complex landscape. Again the plot summarises a single run, there is no averaging of multiple runs.

[1]This value has to be high enough to be above the error threshold for the landscape under study.

```
Procedure Consensus_Plot_Below {
    p = 0.0;     /* Initial mutation rate of 0.0 */
    Initialise the population (at optimum);
    Run_GA;        /* large number of generations */
    Calculate and plot the consensus sequence;
    /* Stop when a predefined (high) mutation rate is reached */
    Until (p = p_max) {
        p = p + p_step;
        Run_GA;    /* large number of generations */
        Calculate and plot the consensus sequence;
    }
}
```

Figure 5.2: Algorithm for producing a consensus sequence plot (from below).

```
Procedure Consensus_Plot_Above {
    p = p_max;     /* Initial (high) mutation rate */
    Initialise the population (randomly);
    Run_GA;        /* large number of generations */
    Calculate and plot the consensus sequence;
    Until (p = 0.0) {
        p = p - p_step;
        Run_GA;    /* large number of generations */
        Calculate and plot the consensus sequence;
    }
}
```

Figure 5.3: Algorithm for producing a consensus sequence plot (from above).

For both approaches, the consensus sequence in the population is calculated and plotted at the end of each simulation cycle for each mutation step. The error threshold is characterised by the loss of the consensus sequence, i.e. the genetic information of the population. Beyond the error threshold the consensus sequence is no longer constant in time (see Figure 5.4).

5.2 Error Thresholds and Evolutionary Parameters

This section uses consensus sequence plots to explore the effect of modifying the values of various evolutionary parameters on the magnitude of error thresholds. Unless otherwise stated, experiments use a generational GA with fitness proportional selection, a population of 100 members, and no recombination, i.e., asexual reproduction. Table 5.1 summarises these default settings. Two instances of landscapes were selected as default test problems: a Royal Staircase function with number of blocks $N = 3$, and block size $K = 10$; and a *NK* landscape[2] with string length $N = 24$, and degree of epistatic interaction $K = 10$. Table 5.2 summarises the default test problems used in most experiments. Further details on the experiments and departures from the default setting are given in the respective subsections.

Population replacement	Generational
Selection Scheme	Proportional
Population size	100
Recombination rate	0.0 (Asexual)
Generations (per mutation rate)	10,000

Table 5.1: GA default parameters used in the experiments.

Landscape	*Setting*	*String length*
NK	$N = 24, K = 10$	24
Royal Staircase	$N = 3, K = 10$	30

Table 5.2: Default test problems used in the experiments.

5.2.1 Preliminary Study

Before exploring the effect of the various evolutionary parameters on the magnitude of error thresholds, three preliminary studies were carried out.

Error thresholds from below and above

The first preliminary study was designed to confirm the assertion by Bonhoeffer and Stadler (1993) that error thresholds do not depend on whether they are approached from below or from above. Figure 5.4 shows the consensus sequence plots on the two default test problems (see Table 5.2). In both cases, the error threshold was approached from below and above (see Section 5.1.1). The Royal Staircase function has a single optimum (the string of all ones). On the other hand, the *NK* landscape has multiple optima, and they are not known. So, for the *NK* landscape the procedure was to first approach the error threshold from above starting from a random population, then store the consensus sequence thus obtained. Afterwards, the error threshold was approached from below starting from a population where all individuals were copies of the stored consensus sequence. In all plots, the vertical axis shows the explored range of mutation values. Mutation rates are expressed as mutations per bit (m/b). Mutation step-sizes were 0.005 for the Royal Staircase and 0.001 for the *NK* landscape.

On both test problems, the plots illustrate the existence of a stable consensus sequence for mutation rates below the error threshold. The consensus sequence is the string of all 1's for the Royal Staircase, and one particular local optima for the *NK* Landscape. Results confirm that error thresholds do not depend on whether they are approached from below or from above. On the *NK*

[2]The *NK* model implementation used throughout this thesis is due to Mitchell Potter, who provides freeware routines in C for generating random *NK* landscapes (http://www.cs.gmu.edu/ mpotter/nk-generator/).

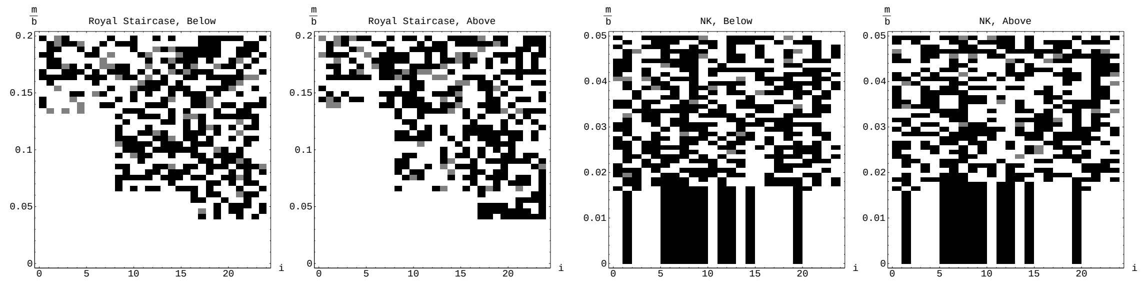


Figure 5.4: Consensus sequence plots on a Royal Staircase function ($N = 3$, $K = 10$) and a NK landscape (N = 24, K = 10). The error threshold is approached from below and above. The horizontal axis shows the consensus bit for each position i, the vertical axis shows per bit mutation rates (m/b). Mutation step-sizes were 0.005 (Royal Staircase), and 0.001 (NK). The error threshold is characterised by the loss of the consensus sequence (the string of all 1's for the Royal Staircase; and one local optima for the NK Landscape). For the Royal Staircase, the intermediate error thresholds for each step or fitness level can also be observed.

landscape the transition occurs close to 0.02 mutations per bit. For the Royal Staircase the critical per bit mutation rate is close to 0.05; in this case, different error thresholds for each fitness level or step can also be observed.

Error thresholds and the initial population

The next set of experiments explores whether error thresholds approached from above (i.e. from a random population) are independent of the initial population. For this purpose, consensus sequence plots are produced for four initial populations (four random seeds) on a fixed NK landscape instance (Figure 5.5). Results suggest that the error threshold is independent of the initial population. Although the consensus sequence achieved in each case is different, the transition occurs at approximately 0.02 mutations per bit in all cases (with a discrepancy of $\simeq 0.002$)

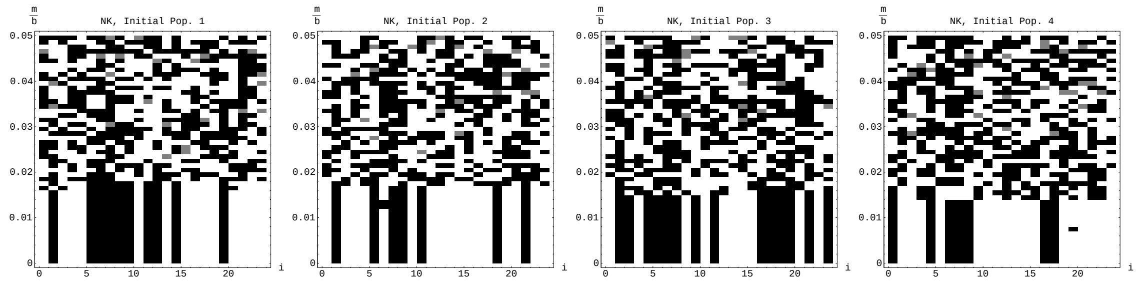


Figure 5.5: Consensus sequence plots on a fixed NK landscape ($N = 24$, $K = 10$) for four initial population (different random seeds). The horizontal axis shows the consensus bit for each position i, the vertical axis shows the mutation rate per bit (m/b). The error threshold was approached from above, that is from a random population. Error thresholds are characterised by the loss of a consensus sequence, which is different in each case.

Error thresholds and landscape instance

The last set of preliminary runs explores whether the error threshold is similar for different instances of an NK landscape with fixed N and K. For each run, a new landscape was generated (using a different random seed for producing the landscape). The initial population was the same in all runs. Error thresholds were approached from above. Results (Figure 5.6) suggest that error thresholds are similar on the four *NK* landscape instances. The transition occurs at approximately 0.02 mutations per bit in all plots.

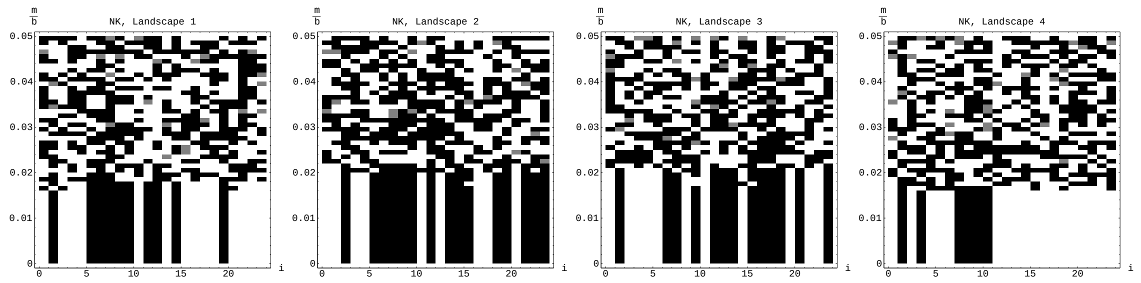


Figure 5.6: Consensus sequence plots on four *NK* landscape instances (different random seeds), all with $N = 24$ and $K = 10$. The initial population was the same in all runs. The vertical axis shows per bit mutation rates (m/b).

5.2.2 Genotype Length

After the preliminary experiments presented above, and for the abstract landscapes and default GA explored, we know that: (i) error thresholds approached from below and above produce similar results, (ii) the error threshold magnitude is independent of the particular initial population. Hence, from now on, experiments will be run approaching the error threshold from above, that is from a random population. Also, a fixed random seed will be used for generating the initial population in all cases. Note as before that approaching error thresholds from above is a more general method, given that it does not require knowing the optimal string beforehand.

The analytical expression of the error threshold on a single peak landscape:

$$p = \frac{ln(\sigma)}{L}, \tag{5.1}$$

suggests that it decreases in proportion to the string length (L). The following experiments explore whether this is also the case on correlated landscapes. Figure 5.7 compares error thresholds on Royal Staircase functions of increasing length. The number of blocks $N = 3$ is kept constant, while the block size is increased from 10 to 12 and 14. Results on the Royal Staircase function suggest that error thresholds (for all fitness levels or steps) decrease as a function of the genotype length. In other words, the longer the genotype the lower the error threshold. This effect is more noticeable for the first and second step transitions.

Figure 5.8 compares consensus sequence plots on *NK* landscapes of increasing genotype length. The parameter $K = 10$ (degree of epistatic interactions) is kept constant, while the string length is varied from 24 to 20 and 28. Results on the *NK* landscape confirm that even small increases in genotype length, decrease the magnitude of the error threshold. The effect is more noticeable when increasing the genotype length from 20 to 24 than from 24 to 28. It should be noticed that the error threshold, if expressed as mutations per string, slightly decreases with each increase in string length, being 0.6 for L = 20, 0.5 for L = 24, and 0.4 for L = 28. These differences may be due to differences in the overall landscape ruggedness. In other words, although K is the

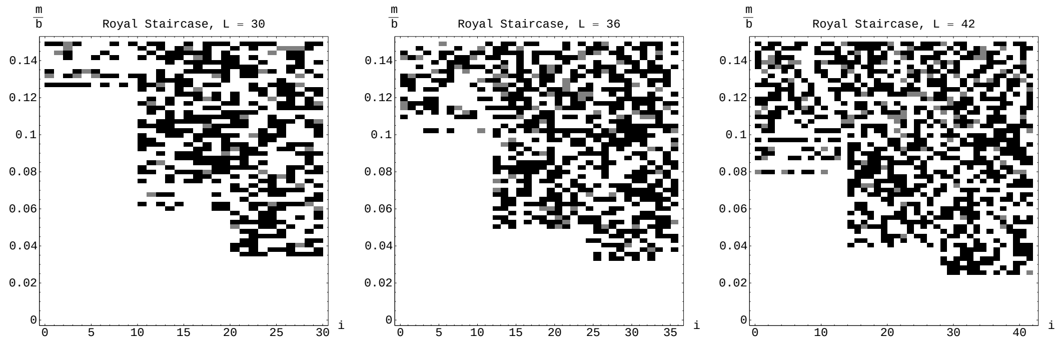


Figure 5.7: Error thresholds and genotype length. Consensus sequence plots on Royal Staircase functions with $N = 3$ and K = 10, 12, and 14 (i.e. string lengths of 30, 36 and 42). The vertical axis shows per bit mutation rates (m/b).

same for all landscapes, the string length (N) varies, which in turns modifies the overall NK landscape structure. This is just a suggested explanation, these results deserve further investigation.

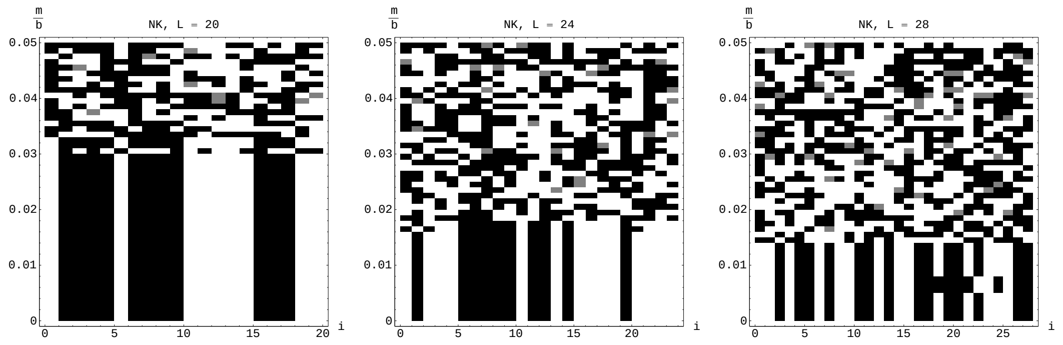


Figure 5.8: Error threshold and genotype length. Consensus sequence plots on NK landscapes with $K = 10$ and $N = L$ = 20, 24, and 28. The vertical axis shows per bit mutation rates (m/b).

From now on, given that error thresholds were shown to depend on the length of genotypes, mutation rates will be expressed as mutations per genotype (m/L where L is the string length) instead of as mutations per bit. Expressing mutation rates per genotype will be more informative when looking for general principles about parameter interactions, since heuristic such as a mutation rate of $1/L$ can be identified.

5.2.3 Selection Pressure

The analytical expression of the error threshold on a single peak landscape (Equation 5.1), suggests that it increases in direct proportion to the strength of selection. The following set of experiments explores whether this is also the case on correlated landscapes. These experiments use tournament selection because this selection scheme allows explicit control over the selection pressure. A common tournament size is 2, but selection pressure increases steadily for growing tournament sizes. Figure 5.9 shows the effect of increasing tournament sizes on the error threshold on both the Royal Staircase and NK landscapes. For both landscapes, the plot using fitness proportional selection is also included for the sake of comparison.

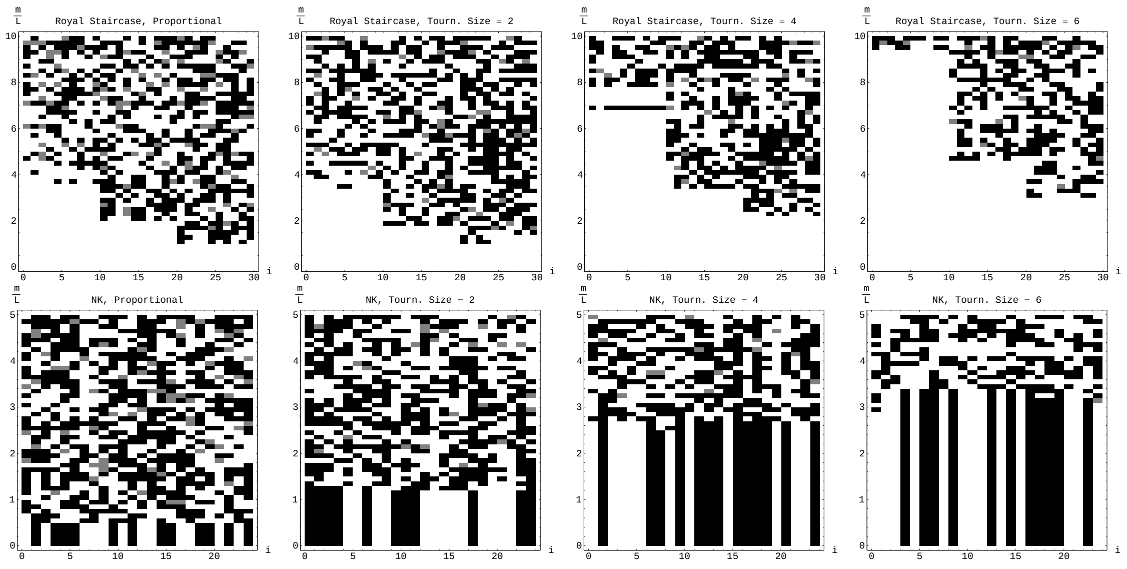


Figure 5.9: Error thresholds and selection pressure. Consensus sequence plots on the selected test problems for tournament sizes of 2, 4, and 6. The plots using proportional selection are also included for the sake of comparison. The vertical axis shows mutation rates per genotype (m/L).

Results on both landscapes show that the strength of selection has a pronounced effect on the error threshold. For increasing tournament sizes (increasing selection pressures) there is a noticeable increase in the magnitude of the error threshold. On the Royal Staircase, the effect is more noticeable for the first and second step transitions. Notice that on the NK landscape (Figure 5.9, bottom), the error threshold for proportional selection is much lower than for tournament selection.

5.2.4 Population Size

This section explores the effect of modifying the population size on the magnitude of error thresholds. The work by Nowak and Schuster (1989), discussed in Chapter 4, extended the calculations of the error threshold on a single peak landscape from infinite to finite populations. Chapter 4 (Section 4.1.2) also shows a reformulation of Nowak and Schuster's analytical expression, which explicitly approximates the extent of the reduction in the error threshold as we move from infinite to finite populations. The expression is an infinite series in which successive terms get smaller; here, only the first few are shown (p_M is the critical rate for a population of size M):

$$p_M = \frac{ln(\sigma)}{L} - \frac{2\sqrt{\sigma-1}}{L\sqrt{M}} + \frac{2ln(\sigma)\sqrt{\sigma-1}}{L^2\sqrt{M}} \tag{5.2}$$

Thus, according to the expression, the error threshold increases with the size of the population given that the second term (the 2nd. greatest of the series) is subtracting and the population size appears in the denominator.

Preliminary Study

As a preliminary study, we compared theoretical error thresholds on a single peak landscape (calculated using Equation 5.2), with empirical error thresholds estimated using consensus sequence plots, on the same landscape for various population sizes. Figure 5.10 shows results of this comparison. The empirical error thresholds were estimated using consensus sequence plots starting from below on a single run. The GA was allowed to run 10,000 generations for each mutation rate (that is, each line of the plot). An acceptable agreement between theory and practice was found. It can be noticed, however, that the difference increases with the size of the population. This may be due to difficulties in reaching the steady-state distribution of the population for higher population sizes. In other words, reaching the steady-state for large populations may require an impractically large number of generations. Differences may also be due to distinct models of evolution. That is, Equation 5.2 was derived using the quasispecies model, whereas empirical error thresholds were estimated using a GA as the underlying model of evolution.

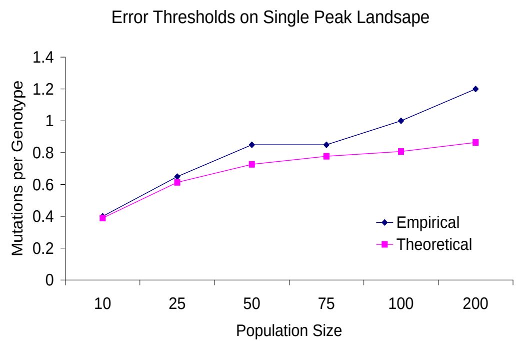


Figure 5.10: Comparing empirical vs. theoretical error thresholds on a single peak landscape for various population sizes.

Main Study

The preliminary study discussed above suggests that the error threshold increases with increasing population size on a single peak landscape. The next step would be, then, to explore whether the same effect is observed on correlated landscapes. Figure 5.11 shows the consensus sequence plots for population sizes of 10, 20, 50, and 100; on the two default test problems.

Results on the Royal Staircase function show that error thresholds (for all fitness levels or steps) increase with increasing population size. The effect is more marked on small populations (sizes 10 and 20), and on error thresholds for the first and second step. Results on the *NK* landscape confirm the increase on error thresholds with increasing population size. Again differences are more noticeable for small populations, and tend to stabilise for larger populations (sizes 50 and 100).

5.2.5 Elitism

The following group of experiments explores the effect of including elitism. Figure 5.12 compares consensus sequence plots with and without elitism on both the Royal Staircase and *NK* landscapes. Two ranges of mutation rates were considered for the elitist version in both cases. Results on the Royal Staircase (Figure 5.12, top) suggest that elitism has a pronounced effect. When elitism

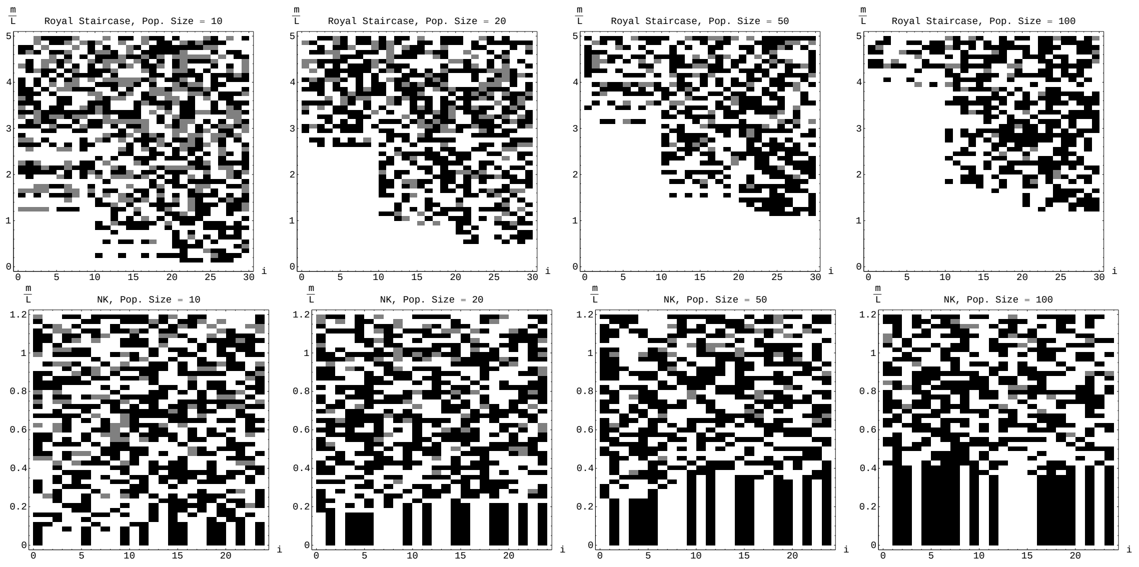


Figure 5.11: Error thresholds and population size. Consensus sequence plots on the selected test problems for population sizes of 10, 20, 50, and 100. The vertical axis shows mutation rates per genotype (m/L).

is used, there is no error threshold transition in the original range of per string mutation rates explored [0.0, 5.0]. If the maximum mutation rate is increased from 5.0 to 20.0 (right plot), a kind of transition is observed around 10.0 mutations per genotype (the pattern of bits becomes more randomised). However, there are no clear transitions for the different fitness levels or steps.

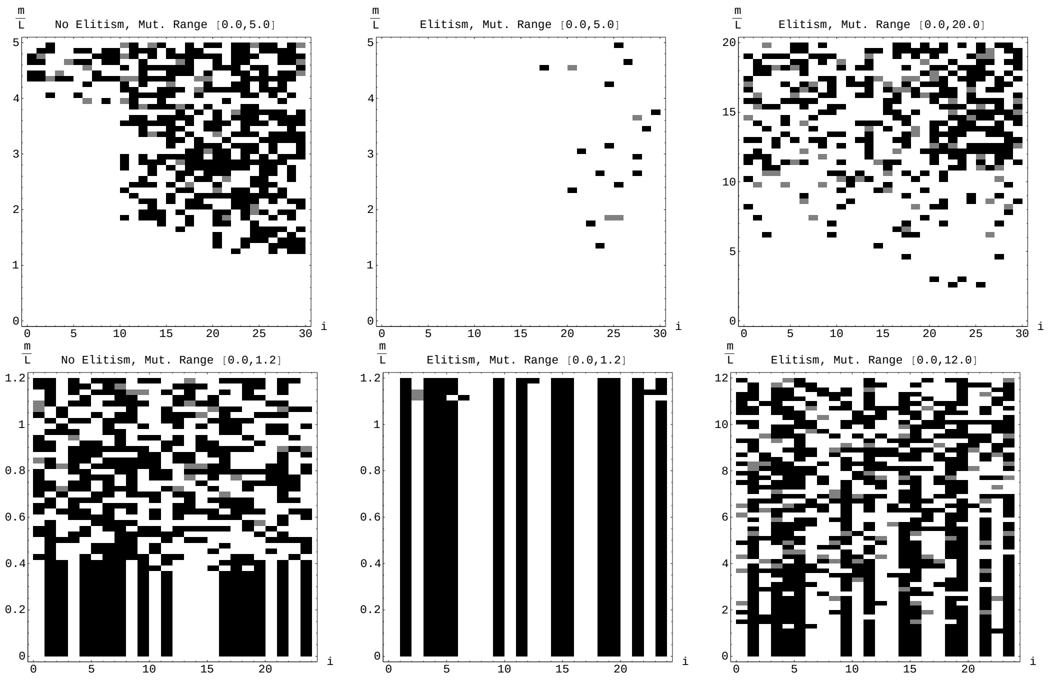


Figure 5.12: Error thresholds and elitism. Consensus sequence plots on the two selected test problems with and without elitism. In both landscapes, the right plot explores a wider range of mutations. The vertical axis shows mutation rates per genotype (m/L).

Results on the NK landscape (Figure 5.12 bottom) confirm the pronounced effect of elitism.

In the first range of mutations explored [0.0,1.2] there is no error threshold transition with elitism. When the maximum mutation is increased to 12.0 (right plot), there is still no clear transition although noise is observed for rates higher than 2.0 mutations per genotype.

5.2.6 Steady State Population Replacement

This set of experiments compares error thresholds using generational and steady-state population replacement. In both cases tournament selection (with tournament size of 2) was used. Three types of steady-state GAs were implemented (see Chapter 2):

1. Using tournament selection for parents, and random selection for individuals that are to be replaced
2. Using random selection for parents, and inverse tournament selection for individuals that are to be replaced
3. Using tournament selection for parents, and inverse tournament selection for individuals that are to be replaced

Figure 5.13 shows the consensus sequence plots for generational replacement and the three types of steady-state replacement discussed above, on the two default test problems. Results on both test problems suggest that error thresholds depend upon the type of steady-state GA used. For type 1, the error threshold is similar to that of generational replacement, although slightly lower. On the other hand, for the other two types of replacement, which include inverse tournament selection for individuals that are to be replaced, the error threshold is noticeably higher (being highest for type 3). This last result is to be expected given that this method imposes the highest selection pressure of the three, since there is selection on parents and individuals that are to be replaced (recall from Section 5.2.3 that error thresholds are higher for higher selection pressures). Following this line of reasoning, results suggest that inverse tournament selection on individuals that are to be replaced, imposes a higher selection pressure than tournament selection on parents. This suggestion is supported by results in evolutionary strategies (Bäck, 1996), which points out that *extinctive* selection (i.e. a selection scheme that definitely excludes some individuals from being selected) imposes a much higher selection pressure as compared to *preservative* selection (i.e. a selection scheme that always assign selection probabilities greater than zero to all individuals). The presence of implicit elitism on steady-state replacement of types 2 and 3, may also accounts for the observed differences on the error thresholds.

5.2.7 Recombination

The work of Boerlijst et al. (1996), and results from Chapter 4 , suggest that recombination shifts the error threshold toward lower values on the single peak and plateau landscapes. The following set of experiments explores whether this is also the case on correlated landscapes. Two types of recombination were considered: 2-point and uniform, both with a rate of 1.0. Figure 5.14 shows the effect of recombination on the Royal Staircase (top) and *NK* landscape (bottom) using 2-point and uniform crossover. For both landscapes the consensus sequence plot without recombination (i.e. crossover rate = 0.0) is included for the sake of comparison.

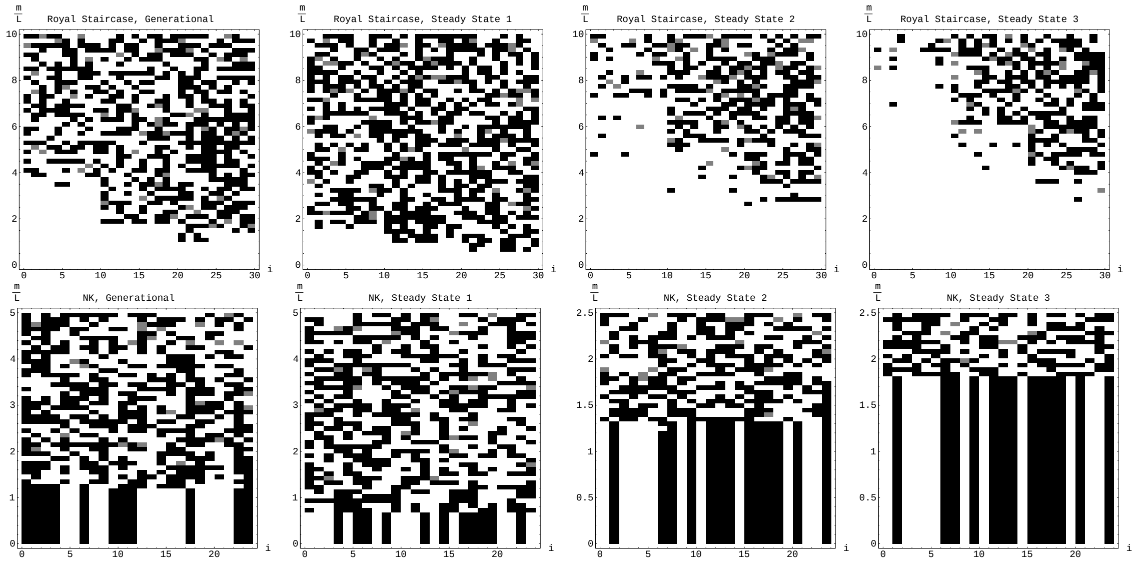


Figure 5.13: Error thresholds and population replacement. Consensus sequence plots on the selected test problems for generational and steady-state population replacement. Three types of steady-state replacement were tested: (1) applying tournament selection on parents and selecting individuals that are to be replaced at random, (2) selecting parents at random and applying inverse tournament selection on individuals that are to be replaced, (3) applying tournament selection on both parents and individuals that are to be replaced. The vertical axis shows mutation rates per genotype (m/L).

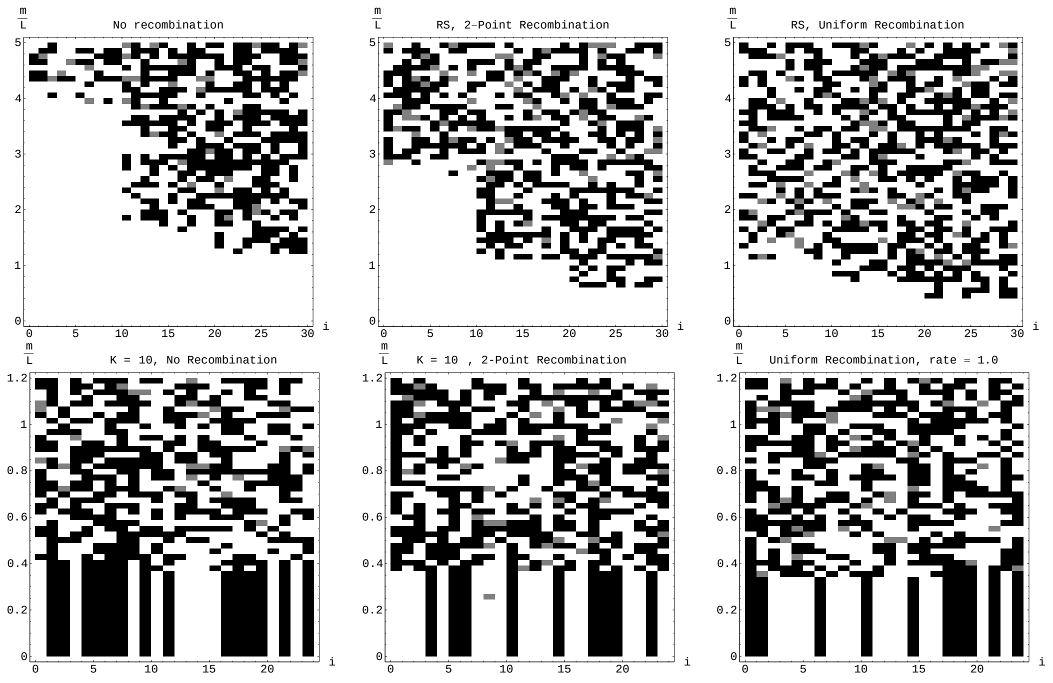


Figure 5.14: Error thresholds and recombination. Consensus sequence plots on the selected test problems with and without recombination. Both two-point and uniform recombination (with a rate of 1.0) were tested. The vertical axis shows mutation rates per genotype (m/L).

On the Royal Staircase function (Figure 5.14, top) error thresholds for all the steps are lower when recombination is used. The plots with no recombination and 2-point recombination are qualitatively similar, whereas the plot using uniform recombination is different in that the transitions for the three steps are closer to one another. On the *NK* landscape with $K = 10$ (Figure 5.14, bottom), there is no noticeable difference in the magnitude of the error threshold with and without recombination. Results from Chapter 4 suggest that the effect of recombination on the error threshold is related to the ruggedness of the landscape. Hence, an extra set of experiments explores the effect of recombination on a *NK* landscape with increased ruggedness ($N = 24$ and $K = 12$). On this new *NK* landscape (Figure 5.15) the error threshold is lower when recombination is used. Results are similar for 2-point and uniform recombination.

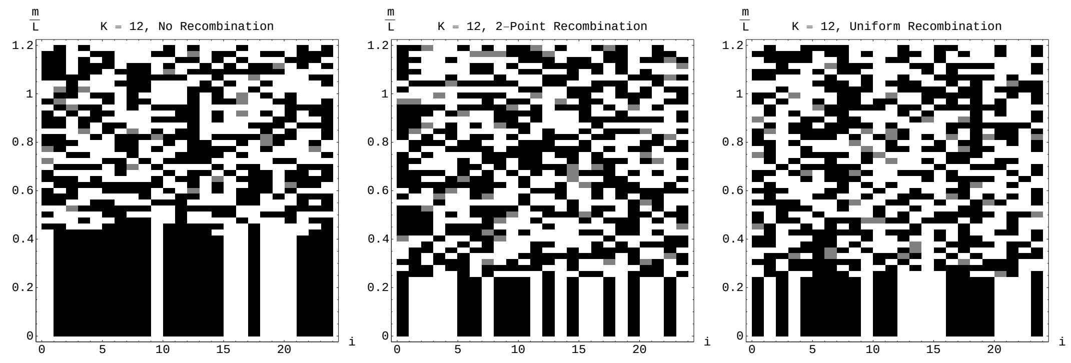


Figure 5.15: Error thresholds and recombination. Consensus sequence plots on a *NK* landscape with increased ruggedness ($K = 12$). Both two-point and uniform recombination (with a rate of 1.0) are tested. The vertical axis shows mutation rates per genotype (m/L).

5.2.8 Assortative Mating

Section 4.3.4 from Chapter 4, explored the effect of assortative mating on the magnitude of error thresholds on the single peak and plateau landscapes. It was found that positive assortative mating, that is when individuals tend to choose mates that are similar to themselves, has the effect of increasing the magnitude of the error threshold. The purpose of this section is, then, to explore whether a similar effect is observed on more complex landscapes. Assortative mating was implemented as in Chapter 4, that is, survival is still based on fitness, but when choosing a partner for a given individual, two potential mates are selected. From this pair, the closest (in Hamming distance) to the first parent, is taken. The recombination operator used was two-point recombination with a rate of 1.0.

Figure 5.16 shows the consensus sequence plots without recombination, recombination with random mating, and recombination with assortative mating, on the two default test problems. Results on both problems confirm that error thresholds are higher when mating is assortative, as compared to both random mating and no recombination. Notice that the increase on the error threshold is similar for both landscapes. Although the range of mutations explored for each landscape is different, the increase on the error threshold magnitude with assortative mating is, in both landscapes, of approximately 0.5 mutations per genotype as compared to no recombination, and about 1.0 mutation per genotype as compared to random mating.

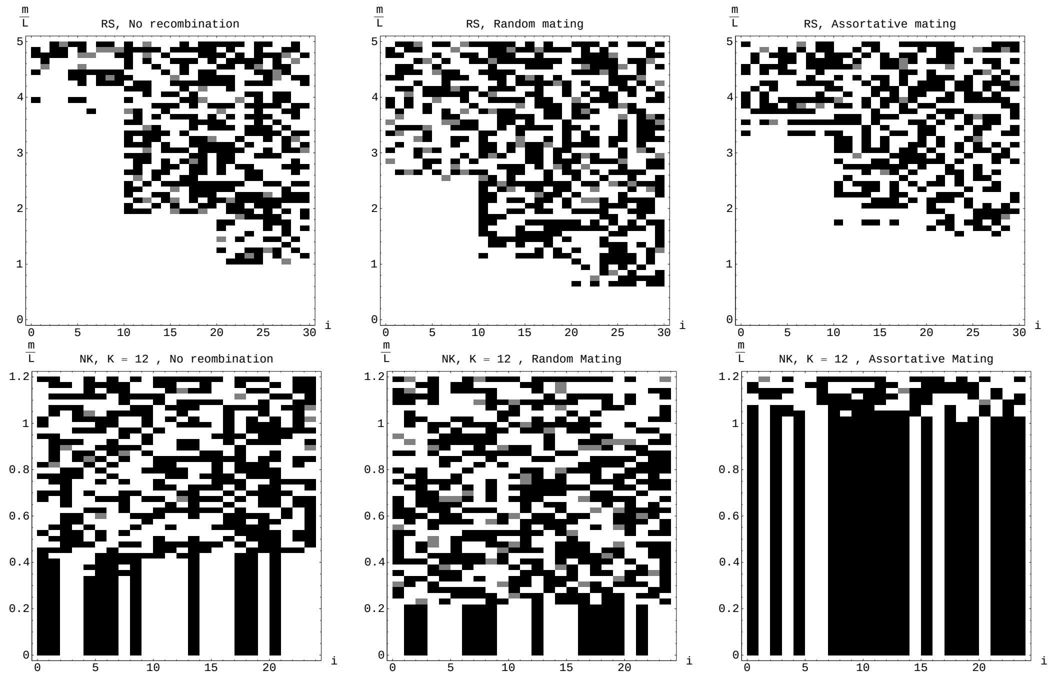


Figure 5.16: Error thresholds and assortative mating. Consensus sequence plots on the selected test problems: without recombination, recombination with random mating, and recombination with assortative mating. Two-point recombination with a rate of 1.0 was used. The vertical axis shows mutation rates per genotype (m/L).

5.2.9 Discussion

This section explored the effect of changing the values of various evolutionary parameters on the magnitude of error thresholds. A few instances of Royal Staircase and *NK* landscapes were used as test problems. The effect of these various evolutionary parameters are summarised below:

- **Genotype length:** Results suggest that error thresholds decrease as a function of the genotype length. In other words, the longer the genotype the lower the error threshold.
- **Selection Pressure:** Results suggest that the strength of selection has a pronounced effect on the error threshold. For increasing selection pressures there is a noticeable increase in the magnitude of error thresholds. Depending on the fitness function, the use of proportional selection may produce much smaller error thresholds as compared to tournament selection.
- **Population Size:** Results show that error thresholds increase with increasing population size. This effect is more marked on small populations (smaller than 50). Differences on the error thresholds stabilise for larger populations; error thresholds for population sizes of 50, 100 and larger are quite similar.
- **Elitism:** Results suggest that elitism has a pronounced effect. When elitism is used, there is no observable error threshold transition. Even if the range of mutations explored is increased, there is no clear transition although some noise is observed.
- **Steady State Population Replacement:** Results suggest that error thresholds depend upon the type of steady-state GA used. When using tournament selection for parents and random selection for individuals that are to be replaced, the error threshold is similar to that of generational replacement. On the other hand, when the steady-state GA includes inverse

tournament selection for individuals that are to be replaced (which is known to impose a higher selection pressure), the error threshold is noticeably higher. These results suggest that the magnitude of the error thresholds depend more on the selection pressure than on the type of replacement. That is, inverse tournament selection on individuals that are to be replaced, imposes a higher selection pressure, which in turn explains the higher error threshold. Also the implicit elitism on some types of steady-state GA accounts for the differences observed on the error threshold magnitudes.

- **Recombination:** For discontinuous functions (Royal Staircase), and very rugged landscapes (NK landscapes with $K > 10$) error thresholds are slightly lower when recombination is used, of the order of 0.2 mutations per genotype. Similar results were obtained for uniform and two-point recombination. However, this effect of recombination was not observed on less rugged landscapes and real-world domains (see Sections 5.3 and 5.4).

- **Assortative Mating:** Results confirm the findings of Chapter 4, that error thresholds are higher when mating is assortative, as compared to both random mating and no recombination. The increase in the error threshold magnitude with assortative mating was of approximately 0.5 mutations per genotype as compared to no recombination, and about 1.0 mutation per genotype as compared to random mating.

5.3 Error Thresholds and Fitness Landscape Structure

This section explores the effect of modifying the landscape structure on the magnitude and characteristics of error thresholds. Hence, we depart from the fixed landscapes used as test problems in the previous section. Instead, various values for the parameters N and K are explored in both the Royal Staircase and NK landscapes. Also, a set of experiments using the NKF family of tunable landscapes (described in Chapter 3, Section 3.2), is presented. All the experiments used a generational GA with tournament selection (tournament size of 2), and a population of size 100. The recombination operator used was two-point recombination with a rate of 1.0 (i.e. a sexual GA). A wide range of mutation rates were explored, mutation rates are expressed as mutations per genotype. Table 5.3 summarises the GA parameter settings used on this set of experiments. Error thresholds were approached from above, that is, starting from a random population and a high mutation rate.

Population replacement	Generational
Selection scheme	Tournament Selection (Tournament Size = 2)
Population size	100
Recombination rate	1.0 (Sexual)
Recombination operator	Two-point
Generations (per mutation rate)	10,000

Table 5.3: GA parameter settings used in the experiments.

5.3.1 Royal Staircase Functions

Royal Staircase functions are always unimodal, but we can increase the function ruggedness by decreasing the number of blocks N. Modifying the number of blocks also alters the overall shape of the landscape. The set of experiments in this subsection maintains a fixed string length of 32, and simultaneously varies *N* and *K*. Table 5.4 summarises the Royal Staircase functions explored.

N	*K*
16	2
8	4
4	8
2	16

Table 5.4: Royal Staircase functions explored.

Figure 5.17 shows the consensus sequence plots on the Royal Staircase functions summarised in Table 5.4. In all functions, the consensus sequence is the single optimum in the landscape (the string of all 1s). In addition, error thresholds for each fitness level or step can be clearly observed. Notice the decreasing number of levels (steps) in the plots as the parameter *N* decreases. For the final step, the error threshold is at approximately 1.5 mutations per genotype for the functions with *N* = 16, 8, and 4, whereas for *N* = 2, the error threshold is at about 1 mutation per genotype. This last observation suggests that error thresholds are lower on very rugged landscapes.

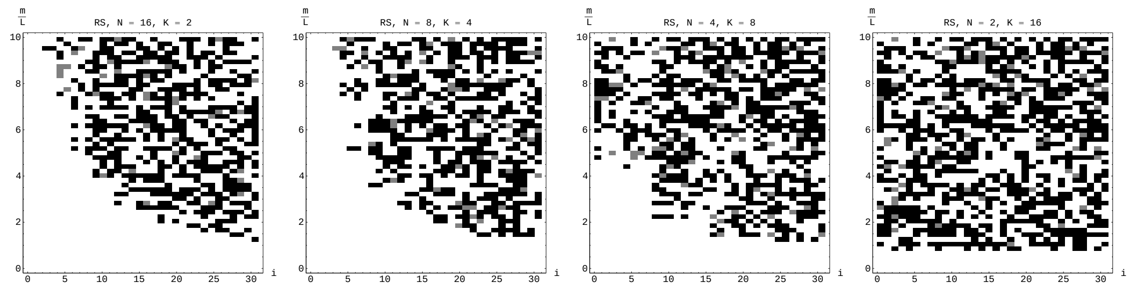


Figure 5.17: Error thresholds on Royal Staircase functions of fixed length and increasing ruggedness. The vertical axis shows mutation rates per genotype (m/L).

5.3.2 *NK* Landscapes

Increasing *K* in the *NK* model increases the landscape ruggedness and number of local optima. Experiments in this subsection use *NK* landscapes with a fixed *N* of 16, and eight values of *K* = {0, 2, 4, 6, 8, 10, 12, 15}. This produces a range of landscapes from a single-peaked and smooth ‘Fujiyama’ landscape (K = 0) to a completely uncorrelated landscape (*K* = 15). Figure 5.18 shows the consensus sequence plots on these landscapes of increasing ruggedness.

The landscapes with low values of *K* (*K* = 0, 2 and 4 in Figure 5.18) show no clear error threshold transition. In fact, the plots show a wide error transition band, and it is in this sense that the error threshold could not be located. On the other hand, the landscapes with medium and high values of *K* (*K* = 6 and greater, in Figure 5.18) show a clear error threshold; there is a distinguishable transition between an “ordered” (selection-dominated) regime, and a “disordered” (mutation-dominated) one. The transition is less clear and sharp on the landscapes with medium ruggedness (*K* = 6 and 8). The higher the value of *K* the sharper the transition and the more

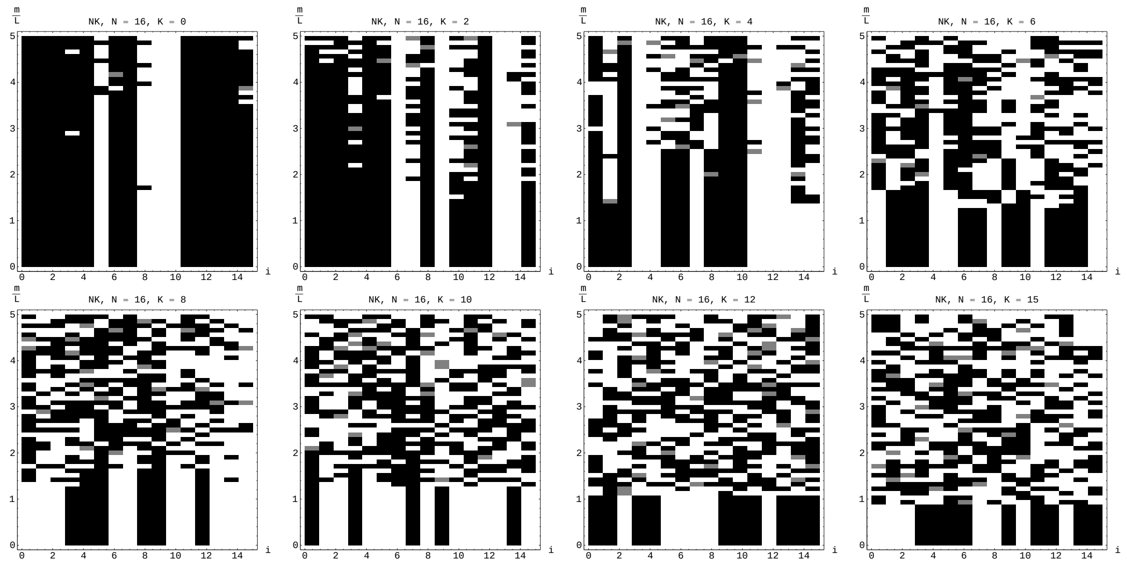


Figure 5.18: Consensus sequence plots on *NK* landscapes of increasing ruggedness. $N = 16$, and $K = \{0, 2, 4, 6, 8, 10, 12, 15\}$. The vertical axis shows mutation rates per genotype (m/L).

"disordered" the pattern of bits beyond the error threshold. Notice also, that the magnitude of the error threshold decreases with increasing K. For $K = 6$ the transition occurs at about 2 mutations per genotype, for $K = 8$ and 10 at about 1.5 mutations per genotype, whereas for $K = 12$ and 15 at about 1.0 mutations per genotype. These results are interesting because they suggest that consensus sequence plots, and the magnitude of error thresholds, say something about the degree of ruggedness of a landscape.

5.3.3 *NKF* Landscapes

The *NKF* model, a generalisation of Kauffman's *NK* landscape (Section 3.2.1), represents a family of landscapes with a tunable degree of neutrality. The parameter F controls the degree of neutrality, which is greatest when F takes the smallest possible value of 2. Experiments in this subsection used *NKF* landscapes with fixed N and K ($N = 24$, $K = 10$), and four values of $F = \{1000, 100, 10, 2\}$. This produces a range of landscapes of increasing neutrality. Figure 5.19 shows the consensus sequence plots for these landscapes. Results suggest that error thresholds remain constant for landscapes of increasing neutrality. It should be noticed, however, that changing F at constant K may alter the overall landscape ruggedness. In all the plots, the transition occurs at approximately 1.5 mutations per genotype.

5.3.4 Discussion

This section explored the effect of modifying the landscape structure on the magnitude and characteristics of error thresholds. For these experiments, GA parameters remained fixed, while landscape parameters were varied. The existence of error thresholds was shown to depend upon the ruggedness of the fitness landscapes. For smooth landscapes, there is no clear error threshold transition. For rugged landscapes, on the other hand, there is a clear transition between an "ordered" (selection-dominated) regime and a "disordered" (mutation-dominated) one. It was found

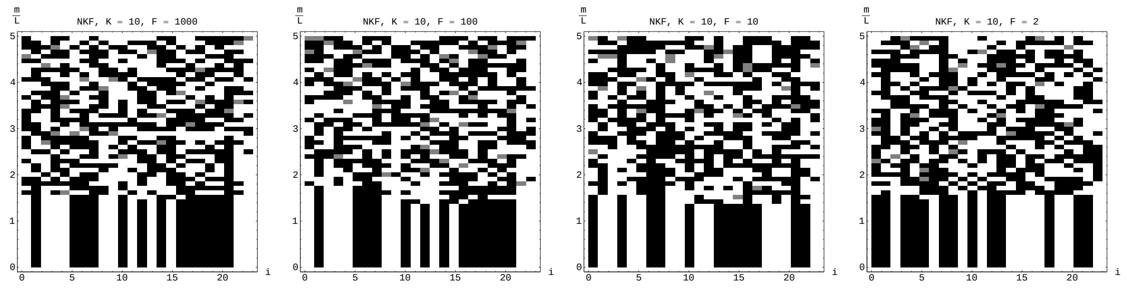


Figure 5.19: Error threshold on NKF landscapes of increasing neutrality. Consensus sequence plots on NKF landscapes of fixed N and K, and varying $F = \{1000, 100, 10, 2\}$. The vertical axis shows mutation rates per genotype (m/L).

that the magnitude of error thresholds decreases with increasing landscape ruggedness. In the range of rugged landscapes studied, the error threshold was located between 1.0 – 2.0 mutations per genotype, being lower for the more rugged landscapes. These results are interesting because they suggest that consensus sequence plots, and the magnitude of error thresholds, say something about the landscape degree of ruggedness. Features such as the presence of steps in Royal Staircase functions were clearly revealed by the consensus sequence plots. Finally, error thresholds were shown to remain unchanged on NKF landscapes of increasing degree of neutrality.

5.4 Error Thresholds in Real-World Domains

The closing empirical section of this chapter explores whether error thresholds can be observed in real-world applications. Two applications were selected: the Wing-Box design optimisation problem, and the Multiple Knapsack problem. For a detailed description of these problems, the reader is referred to Chapter 3, Section 3.2.2.

All the experiments used a generational GA with tournament selection (tournament size of 2), and a population of size 100. The GA was run in two modes: (i) using mutation only (*Asexual*), and (ii) using both mutation and recombination (*Sexual*). The recombination operator used was two-point recombination with a rate of 1.0. The mutation rate range explored was from 0.0 to 5.0 mutations per genotype with a step of 0.1. Each simulation cycle lasted 15,000 generations. Error thresholds were approached from above, that is, starting from a random population and a high mutation rate.

5.4.1 Wing-Box Problem

Two groups of Wing-Box experiments were run. First, using the redundant encoding described in Chapter 3 (see Table 3.4). Second, using the non-redundant mapping also described in Chapter 3 (see Table 3.5). For all the experiments the number of panels was set to 50. Recall from Chapter 3 that the encoding of the Wing-box problem requires 13 bits for coding the absolute thickness of the first panel, and 3 bits for encoding the relative increment/decrement in thickness of the remaining panels. So, the number of bits needed for encoding an individual is 13 for the first panel, and 3 for each of the others 49 panels, that is $13 + 3 \times 49 = 160$.

Figure 5.20 shows results on the Wing-Box problem for asexual and sexual GAs, using both

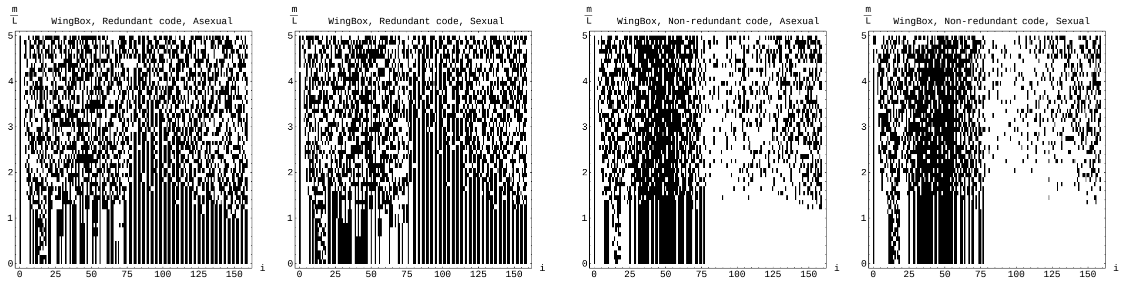


Figure 5.20: Error thresholds on two encodings of the Wing-Box problem for both asexual and sexual GAs. The vertical axis shows mutation rates per genotype (m/L).

the redundant and non-redundant encodings. The plots show the existence of a stable consensus sequence for mutation rates below the error threshold. The error threshold is visualised as the transition from a stable consensus sequence to a random sequence of bits. Notice that, for the redundant mapping (left plots), the error threshold transition is not clear. For most of the bits the transition occurs somewhere between 1.0 and 2.0 mutations per genotype, but from approximately bit 75 to bit 125 the error threshold looks higher. On the other hand, the non-redundant encoding (right plots) shows a clearer transition, which for most of the bits occurs around 1.5 mutations per genotype. An exception is the portion of bits from 11 to 16, which are randomised even for low mutation rates. These bits correspond to the less significant digits of the thickness of the first panel, and the relative thickness of the second panel. Given the characteristics of the problem, these bits are neutral in that changes to them are not reflected in the overall fitness of the wing-structure. Also, on the non-redundant mapping, there is still a region from bit 75 to bit 125 where the consensus sequence seems more stable for higher mutation rates.

Notice that the consensus sequence plots are giving us some information about the solutions found. For the redundant coding, ‘100’ encodes the relative thickness increment of -0.25, whereas the non-redundant code represents this value with ‘111’ (see Tables 3.4 and 3.5 from Chapter 3). In all plots, and below the error threshold, this value (-0.25) is fixed from bit 75 onwards, that is from panel 21 onwards.

Finally, for both encodings, there are no clear differences between the asexual and sexual GA regarding the magnitude of error thresholds.

5.4.2 Multiple Knapsack Problem

Four multiple-knapsack instances, taken from the literature, were selected as test problems. Problem sizes ranged from 50 to 105 objects and from 2 to 30 knapsacks. Table 5.5 summarises the problem instances tested. These (and several other) problems are available online from the OR-library by Beasley (1990).

Figure 5.21 shows the consensus sequence plots on the four Knapsack instances selected, asexual (top) and sexual (bottom). Results on the four instances confirm the existence of error thresholds on this real-world application. The error threshold is again visualised as the transition from a stable consensus sequence to a more randomised sequence of bits. The transition in all the instances occurs at approximately 1.0 – 1.2 mutations per genotype. Notice that in all instances there are some regions of the genotype where the consensus sequence is more stable for mutation

Instance	*Objects*	*Sacks*
Weish 12	50	5
Sento 1	60	30
Weing 7	105	2
Weish 30	90	5

Table 5.5: Multiple Knapsack problem instances tested.

rates beyond the error threshold. There are no clear differences between the GA with and without recombination regarding the magnitude of error thresholds. However, the transitions looks sharper, and thus the consensus sequences less stable, for the GA without recombination (asexual). This may be due to the use of two-point recombination. It is known that two-point recombination is a less disruptive operator than mutation alone or uniform recombination.

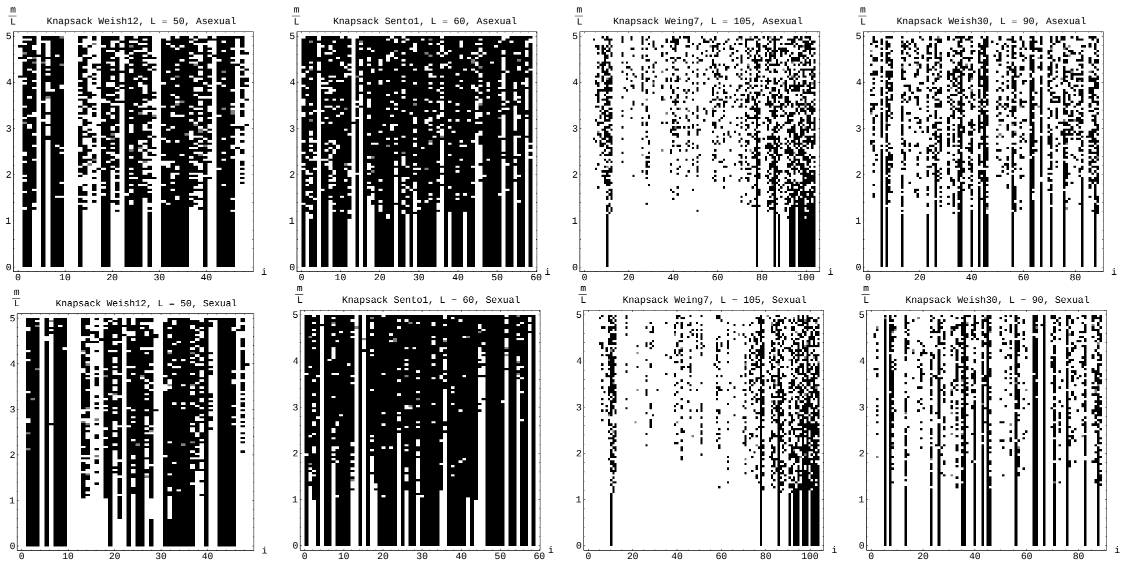


Figure 5.21: Error thresholds on four instances of the Multiple Knapsack problem. Results for asexual (top) and sexual (bottom) GAs are presented. The vertical axis shows mutation rates per genotype (m/L).

5.4.3 Discussion

This closing empirical section explored error thresholds on real-world domains. Results show that error thresholds can also be found on these two complex real-world applications. It should be noticed, however, that other real-world applications might have very different characteristics. No major differences were noticed in the magnitude of error thresholds on GAs with and without recombination. In all scenarios, for the particular GA selected: tournament selection (tournament size of 2), population size of 100, and generational replacement, the error threshold was located at approximately 1.0 – 1.5 mutations per genotype.

5.5 Conclusions

This chapter verifies the occurrence of error thresholds in evolving populations of bit strings using a GA (with and without recombination). Error thresholds were observed on several landscapes, including real-world domains. In this way, the notion of error threshold, (already introduced for very simple landscapes in Chapter 4) is brought to evolutionary computation.

This chapter also introduced the *consensus sequence plots*. These plots, borrowed and adapted from theoretical biology (Bonhoeffer & Stadler, 1993), are new to the evolutionary computation community. They represent a novel way to visualise the structure of fitness landscapes, since features such as the presence of steps or discontinuities can be noticed. Moreover, the degree of ruggedness in a landscape was revealed by these plots. Consensus sequence plots may also serve as a tool to differentiate critical (and less critical) areas in the genotype, which may have practical implications when tackling real-world problems. First, it may be possible to infer important knowledge about an applied problem. Second, it may be possible to refine the genotype representations and optimal schedules for mutation rates. This may be possible on some classes of problems, as for instance the Wing-Box and Knapsack problems, where producing the consensus sequence plot took few hours (on a standard Sun SPARC Station). However, consensus sequence plots are computationally expensive and may be infeasible for other present-day challenging problems.

The next chapter will explore the hypothesised relationship between error thresholds and optimal mutation rates. It will also explore the effect of modifying both the values of evolutionary parameters and the fitness landscape structure, on the magnitude of optimal mutation rates. The major lesson learned from this chapter is that error thresholds depend mainly on the selection pressure and the genotype length, regardless of the landscape under study, as long as the landscape is rugged. This knowledge may suggest useful heuristics for setting near-optimal mutation rates. In particular, the suggestion of setting a mutation rate of $1/L$ (one mutation per genotype), discussed in Chapter 2, is supported by the experiments in this chapter, but only on rugged landscapes, population sizes greater than 50, and selection schemes imposing a selection pressure similar to that of tournament selection with a tournament size of 2. The $1/L$ heuristic is most probably applicable on landscapes with little or no redundancy. As suggested by Harvey and Thompson (1996), in the presence of redundancy or ‘junk’ this heuristic should be adjusted so as to give an expected 1 mutation per *non-redundant* part of the genotype. These ideas will be further explored in the next chapter.

Chapter 6

Optimal Mutation Rates in Genetic Algorithms

The previous chapter demonstrated the occurrence of error thresholds in GAs running on a wide range of fitness landscape topologies, including real-world domains. It also explored the effects of modifying both the settings of evolutionary parameters, and the structure of landscapes, on the magnitude of error thresholds. This chapter continues with a similar study but explores optimal mutation rates instead of error thresholds, and preferentially uses real-world domains as test problems. It also studies the relationship between error thresholds and optimal mutation rates by comparing these two measures over various problems.

The chapter is organised as follows. The method section discusses the working definition of an 'optimal' mutation rate, and describes the empirical approach used here for estimating it. Section 6.2 studies the relationship between error thresholds and optimal mutation rates, by comparing these two measures on both abstract problems and real-world domains. Thereafter, Section 6.3 uses two instances of a real-world problem (the Multiple Knapsack) to explore the effect of varying the most relevant evolutionary parameters on the magnitude of optimal mutation rates. The closing empirical section of the chapter (Section 6.4) uses a fixed GA (fixed evolutionary parameters) and various parameter settings of the tunable abstract landscapes described in Chapter 3 (Section 3.2.1), to explore the effect of modifying the landscape structure on the magnitude and range of optimal mutation rates.

6.1 Method

For estimating optimal mutation rates in GAs we need to define what an optimal or near-optimal mutation rate is. The working definition used here is: an optimal mutation rate is that producing optimal performance. But then, we need a good way of measuring GA performance. Recall from Chapter 2 (Section 2.2.8), that given the randomised nature of GAs, conclusions can never be drawn from a single run. Instead, the common practice is to consider statistics from a sufficiently large number of independent runs. So, the standard performance measures for GAs are the average and best fitness values attained after a prefixed termination criterion, averaged over several runs. Within a given run, the best fitness could be either the current best in the population, or the best fitness attained so far. These measures are considered after a fixed termination criterion, or over fixed intervals throughout the GA run. For the experiments in this chapter, we will consider the best fitness attained so far after a fixed termination criterion. This criterion will be carefully selected in each case to be long enough to stabilise the best and average fitness of the population. The average of several runs will be considered (typically 50) and the standard deviation will be shown in most cases.

6.2 Optimal Mutation Rates and Error Thresholds

This section explores the relationship between error thresholds and optimal mutation rates. The approach followed is to independently assess these two measures and compare them. Error thresholds were estimated already on several landscapes in Chapter 5, so these values will be used whenever possible. Unless otherwise stated, the experiments for estimating optimal mutation rates use a generational GA with tournament selection (tournament size of 2), and a population of size 100. The GA is run in two modes, using mutation only (Asexual), and using both mutation and recombination (Sexual). The recombination operator used is two-point recombination with a rate of 1.0. A wide range of mutation rates are explored, they are expressed as mutations per genotype.

6.2.1 Preliminary Study: Termination Criteria (Generations vs. Evaluations)

As mentioned in the method section, the approach followed here for measuring GA performance is to calculate the average best-so-far fitness attained after a fixed termination criteria (a fixed number of generations or function evaluations). The termination criterion is carefully selected to allow the population to equilibrate its average and best fitness. This section explores whether the choice of (i) a fixed number of generations, or (ii) a fixed number of evaluations, makes a difference when estimating optimal mutation rates. When considering the number of evaluations, a simple optimisation can be performed (at least for deterministic fitness functions), such that only newly created individuals need to be evaluated. In other words, if an individual passed without modifications to the following generation, there is no need to re-evaluate it. Instead, its fitness value is maintained. From now on, when referring to number of evaluations, it is considered that evaluations are counted in this manner, that is, only considering newly created individuals.

Experiments in this section explore optimal mutation rates on a Royal Staircase function with $N = 3$ and $K = 10$. Two termination criteria were tested: (i) a fixed number of generations (1,000) and (ii) a fixed number of function evaluations (100,000 for the sexual GA, and 40,000 the asexual GA). Figure 6.1 shows results using tournament selection (tournament size = 2), whereas Figure 6.2 shows results using proportional selection. In the figures, each point is the average of 50 runs, error bars show $\pm$ the standard deviation. The Royal Staircase is a maximisation problem, so optimal mutation rates are those producing the highest average best-so-far fitness.

With tournament selection with and without recombination (Figure 6.1), optimal mutation rates were similar for both termination criteria. On the other hand, with proportional selection (Figure 6.2), optimal mutation rates were similar for both termination criteria for the GA with recombination (sexual). However, with no recombination and considering evaluations, the results were different than those considering generations. Specifically, optimal mutation rates tended to be lower and close to zero when considering evaluations (Figure 6.2, bottom right plot). These anomalous results deserve further investigation.

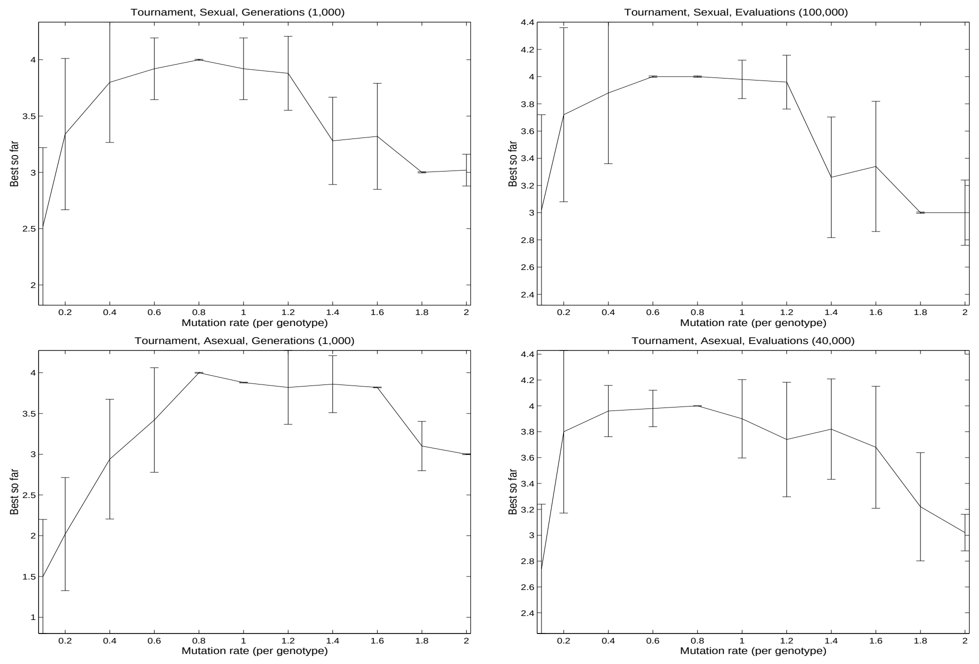


Figure 6.1: Optimal mutation rates (per genotype) on a Royal Staircase function ($N = 3$, $K = 10$) using tournament selection (T. size = 2) and considering two termination criteria. Left: a fixed number of generations (1,000). Right: a fixed number of function evaluations (100,000 for the sexual GA, and 40,000 for the asexual GA). The curves show the average best-so-far fitness attained after the termination criterion is reached for various mutation rates.

Notice that, for both selection schemes, while the same number of generations were needed for the sexual and asexual GA to equilibrate the average best-so-far fitness (1,000 generations), many more functions evaluations were needed for the sexual GA (100,000) as compared to the asexual (40,000) to reach this stage. This last observation, together with the anomalous results with proportional selection and no recombination reported above, suggest that, at least for this kind of very neutral landscape, a mutation-only algorithm with a low mutation rate may produce better performance than a standard GA. This deserves further investigation but goes beyond the scope of this dissertation.

Discussion

When estimating optimal mutation rates with tournament selection, it was found that the choice of termination criteria, (i) a fixed number of generations or (ii) a fixed number of function evaluations, does not noticeably affect the results. Some anomalous results were, however, found with proportional selection and no recombination when considering function evaluations. This may lead to potentially interesting research. Following the results from this preliminary study, it was decided to use tournament selection and a fixed number of generations as the termination criterion for all the remaining experiments in this chapter.

6.2.2 Royal Staircase Function

Table 6.1 compares error thresholds, as estimated in Chapter 5, and optimal mutation rates as estimated above on the Royal Staircase function with $N = 3$ and $K = 10$ (Figures 6.1 and 6.2).

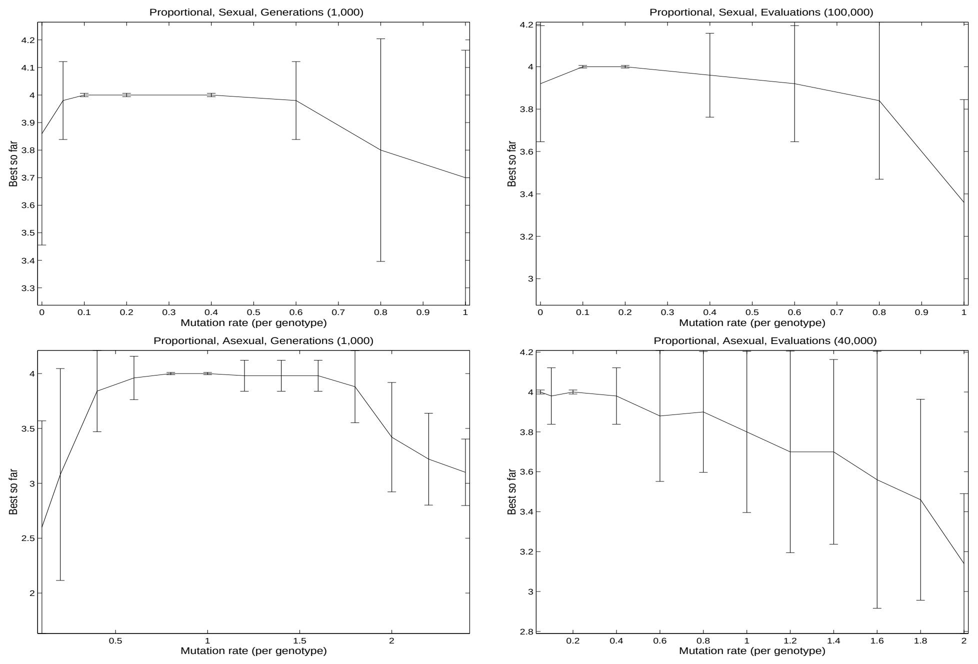


Figure 6.2: Optimal mutation rates (per genotype) on a Royal Staircase function ($N = 3$, $K = 10$) using proportional selection and considering two termination criteria. Left: a fixed number of generations (1,000). Right: a fixed number of function evaluations (100,000 for the sexual GA, and 40,000 for the asexual GA). The curves show the average best-so-far fitness attained after the termination criterion is reached for various mutation rates.

Results for both tournament selection (tournament size = 2) and proportional selection are shown. All these measures are empirical approximations, and are expressed as mutations per genotype. Optimal mutation rates were found to be a less precise measure as compared to error thresholds, so a range of optimal mutation values instead of a single value is presented.

	Tournament Selection		Proportional Selection	
	Opt. Mut. Rates	Error Threshold	Opt. Mut. Rates	Error Threshold
Sexual GA	0.6 – 1.2	0.9	0.1 – 0.6.	0.6
Asexual GA	0.8 – 1.6	1.0	0.6 – 1.8	1.1

Table 6.1: Comparing optimal mutation rates (per genotype) and error thresholds on a Royal Staircase function with $N = 3$, and $K = 10$, for both tournament and proportional selection.

Notice that (Table 6.1) error thresholds are located within the range of optimal mutation rates in all cases. Moreover, recombination has a similar effect on both error thresholds and optimal mutation rates, namely, to shift them to lower values as compared to no recombination. This effect is more noticeable for proportional selection.

6.2.3 Multiple Knapsack Problem

Two Multiple Knapsack instances, taken from the literature, are used as test problems, Table 6.2 summarises them. These (and several other) problems are available online from the OR-library by Beasley (1990). The instances selected are among the biggest and more complex available in the library.

Instance	*Objects*	*Sacks*
Sento 1	60	30
Weish 30	90	5

Table 6.2: Multiple Knapsack problem instances tested, with the names they're referred to on the OR-library.

On the two selected instances (Figure 6.3), the curves show the average best-so-far fitness attained after 3,000 generations for various mutation rates. In the figures, each point is the average of 50 runs, error bars show $\pm$ the standard deviation. In all cases, an optimal mutation rate or range of optimal mutation rates could be identified. This is a maximisation problem, so optimal mutation rates are those producing the highest average best-so-far fitness. In all cases, error thresholds (indicated in the plots) are located within or close to the range of optimal mutation rates.

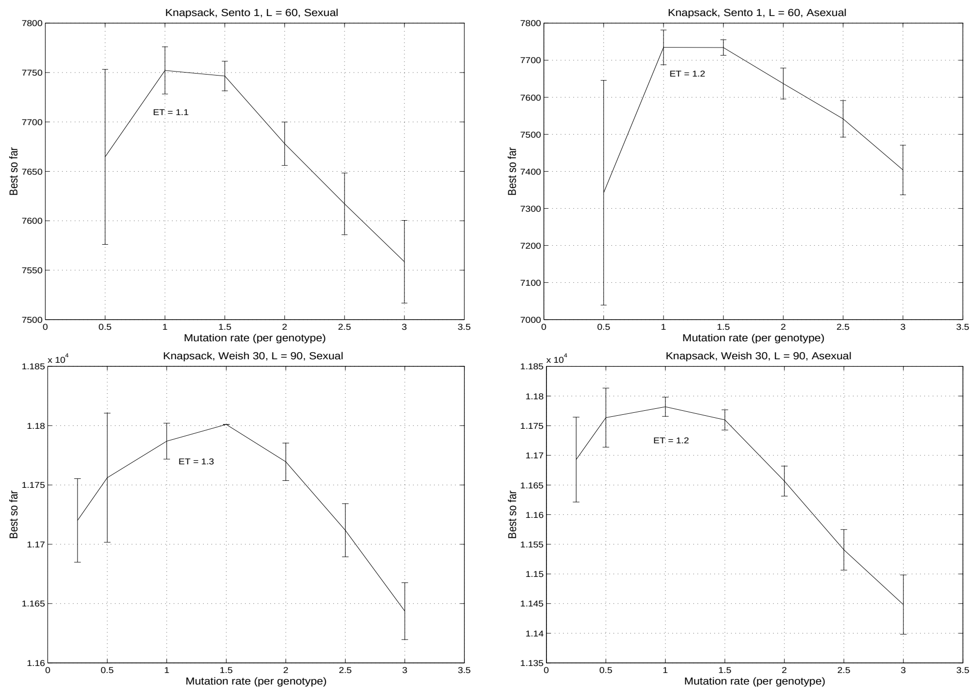


Figure 6.3: Optimal mutation rates on two instances of the Multiple Knapsack problem. The curves show the average best-so-far fitness attained after 3,000 generations for various mutation rates. Results with (right) and without (left) recombination are shown.

Figure 6.4 compares the algorithm performance with and without recombination on the two selected instances for various mutation rates. Error bars are not shown for the sake of clarity. Results suggests that using recombination improves the average best-so-far fitness. On the instance named Sento 1, the improvement is observed over all the mutation rates explored. On Weish 30, performance was similar for the lowest mutation rates, but better for the sexual GA for mutation rates higher than $1.0/L$. Moreover, the highest average best-so-far fitness was attained using recombination and a mutation rate of $1.5/L$.

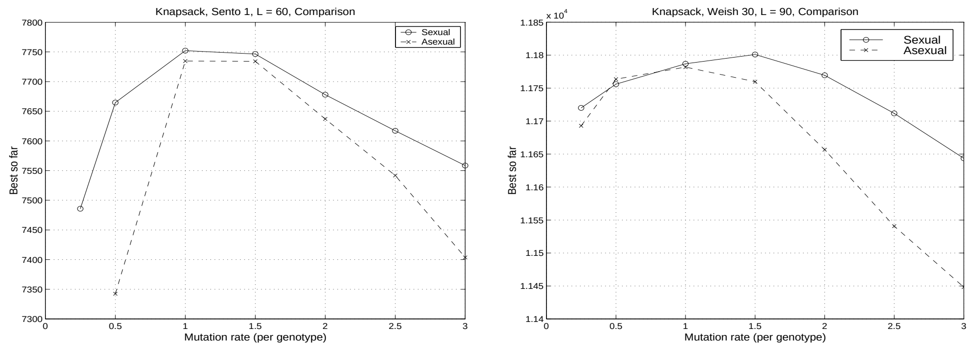


Figure 6.4: Comparing performance with (sexual) and without recombination (asexual) on two instances of the Multiple Knapsack problem. Average best-so-far fitness attained after 3,000 generations for various mutation rates.

6.2.4 Wing-Box Problem

This subsection explores two encodings of the Wing-Box problem: a non-redundant and a redundant encoding (see Chapter 3, Section 3.2.2). Recall from Chapter 3 that the encoding of the Wing-Box problem requires 13 bits for coding the absolute thickness of the first panel, and 3 bits for encoding the relative increment/decrement in thickness of the remaining 49 panels. So, the number of bits needed for encoding an individual is 13 for the first panel, and 3 for each of the others 49 panels, that is $13 + 3 \times 49 = 160$. Figure 6.5 shows results on the Wing-Box problem with and without recombination, using both the redundant and non-redundant encodings. The curves show the average best-so-far fitness attained after 3,000 generations for various mutation rates. Each point in the curves is the average of 50 runs, error bars show $\pm$ the standard deviation. Since this is a minimisation problem, optimal mutation rates are those producing the minimal average fitness. Error thresholds are indicated in the plots, notice that they are located near the estimated optimal mutation rates in all cases. Note also that optimal mutation rates are higher on the redundant coding (bottom plots), being around $2.0/L$.

Figure 6.6 compares the algorithm performance with and without recombination on the two encodings of the Wing-Box problem for various mutation rates. Error bars are not shown for the sake of clarity. On the non-redundant encoding, average best-so-far fitness was similar for low mutation rates ($\leq$ 1.0/L), but better for the sexual GA for higher mutation rates. On the redundant encoding, the sexual GA produced better performance than the asexual GA over all the mutation rates explored.

6.2.5 Discussion

Results on the Royal Staircase functions suggested that there is no single optimum mutation rate, but instead a range of values producing near-optimal performance. On the other hand, the real-world domains showed a more definite optimal mutation rate or at least a smaller range of near-optimal mutation values. This feature of Royal Staircase functions may be due their characteristic high degree of neutrality. It was found that, on the Royal Staircase function, error thresholds are located within the range of optimal mutation rates. It was also found that recombination shifts optimal mutation rates to lower values, which mirrors the effect of recombination on error

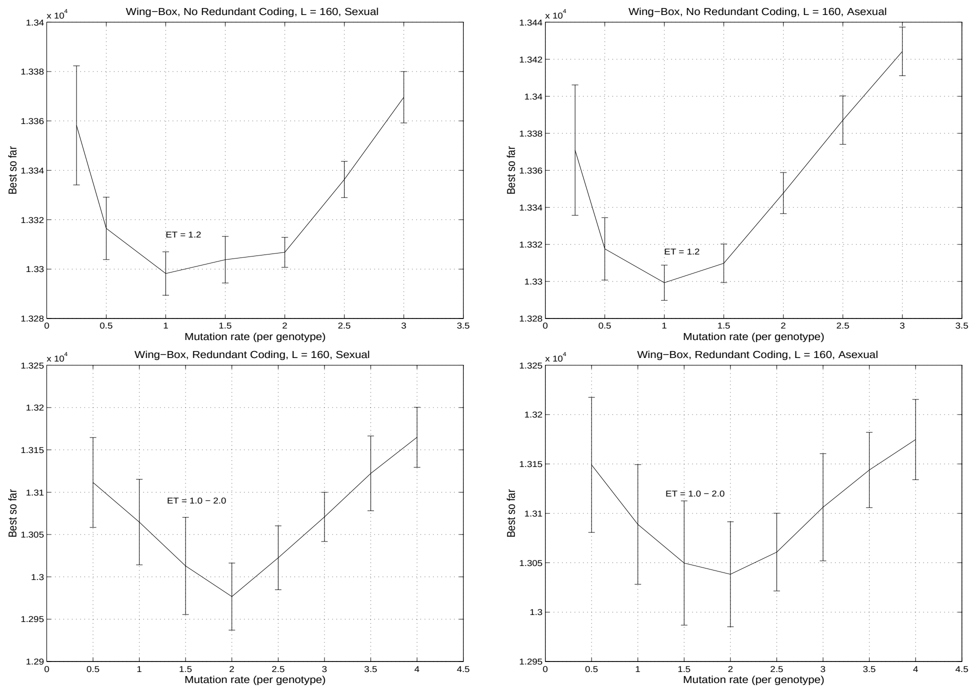


Figure 6.5: Optimal mutation rates on two encodings of the Wing-Box problem. The curves show the average best-so-far fitness attained after 3,000 generations for various mutation rates. Since this is a minimisation problem, optimal mutation rates are those producing the minimal average fitness. Results with (sexual, left) and without (asexual, right) recombination are shown.

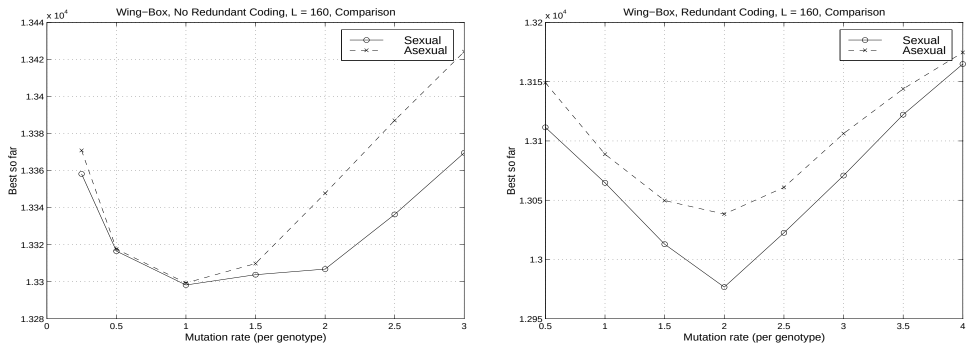


Figure 6.6: Comparing performance with and without recombination on two encodings of the Wing-Box problem. Average best so far fitness attained after 3,000 generations for various mutation rates.

thresholds on this landscape.

On the real-world problems, optimal mutation rates with and without recombination are in the same range. If the optimal mutation rate is selected, the GA with recombination generally produces highest average best-so-far fitness. In most scenarios, for the particular GA selected: tournament selection (tournament size of 2), population size of 100, and generational replacement; optimal mutation rates were around 1.0 – 1.5 mutations per genotype, which corresponds to the magnitude of error thresholds in these domains as estimated in Chapter 5. An exception to this behaviour was observed for a redundant genetic encoding where the optimal mutation rate was slightly higher: around 2.0 mutations per genotype.

The empirical evidence in this section suggests a correlation between error thresholds and optimal mutation rates. Moreover, this relationship carried over from a simple toy-problem such as the Royal Staircase function to complex real-world applications.

6.3 Optimal Mutation Rates and Evolutionary Parameters

This section explores the effect of modifying the most relevant evolutionary parameters on the magnitude of optimal mutation rates. Unless otherwise stated, experiments use a generational GA with tournament selection (tournament size = 2), a population of 100 members, and both mutation and recombination (two-point with a rate of 1.0), i.e. a sexual GA. Table 6.3 summarises these default settings. This chapter emphasises the use of real-world domains as test problems, thus the two Multiple Knapsack problem instances summarised in Table 6.2 (Section 6.2.3) were used. Further details on the experiments and departures from the default settings are given in the respective subsections. The approach for estimating optimal mutation rates is to calculate the average (of 50 runs) best-so-far fitness attained after 3,000 generations for several mutation rates.

Population replacement	Generational
Selection scheme	Tournament (T. Size = 2)
Population size	100
Recombination rate	1.0 (Sexual)
Recombination operator	Two-point
Termination criterion	3,000 Generations
Number of runs	50

Table 6.3: GA default parameters used in the experiments.

6.3.1 Genotype Length

Experiments in this subsection attempt to explore the effect of modifying the genotype length on the magnitude of optimal mutation rates. The selected Knapsack problem instances, Sento1 and Weish 30, have string lengths of 60 and 90 respectively. Figure 6.7 shows the average best-so-far fitness attained after 3,000 generations on these instances for various mutation rates, expressed as mutations per bit. Although these are different problems and definitive conclusions cannot be drawn, results suggest that the optimal mutation rate is lower (0.015 mutations per bit) for the longer genotype.

6.3.2 Selection Pressure

This subsection explores the effect of increasing the selection pressure on the magnitude of optimal mutation rates. The experiments use tournament selection because this scheme allows the selection pressure to be explicitly controlled. A common tournament size is 2, but selection pressure increases steadily for growing tournament sizes. Two tournament sizes, 2 and 4, were tested. Additionally, in one of the instances: Weish 30, results using proportional selection are also pre-

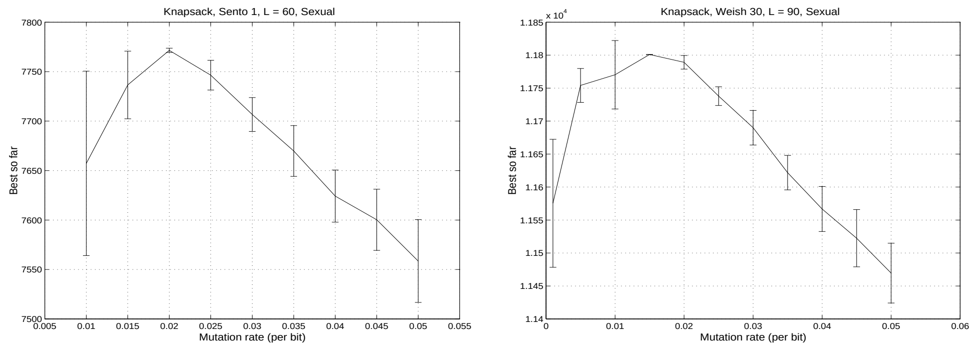


Figure 6.7: Comparing (per bit) optimal mutation rates on Multiple Knapsack instances of different string lengths (L = 60, 90). The curves show the average best-so-far fitness attained after 3,000 generations for various mutation rates.

sented for the sake of comparison[1]. Figure 6.8 compares optimal mutation rates (per genotype) on the two selected problem instances. The strength of selection had a pronounced effect on the magnitude of optimal mutation rates: for a tournament size of 2, the optimal mutation rate was 1 – 1.5 mutations per genotype, whereas for a tournament size of 4 it was 2.5 – 3.0 mutations per genotype. Moreover, the curve using proportional selection on Weish 30 (Figure 6.8, right), strikingly shows the difference in magnitude of optimal mutation rates for a weak selection pressure. In this case, the optimal mutation rate was as low as 0.05 mutations per genotype.

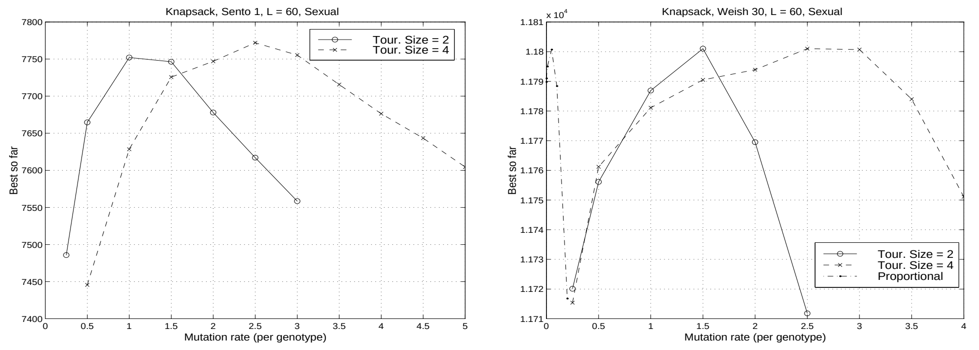


Figure 6.8: Comparing optimal mutation rates (per genotype) for different selection pressures on two instances of the Multiple Knapsack problem. Tournament selection with two tournament sizes (2 and 4) was tested. Additionally, proportional selection was tested on Weish 30. The curves show the average best-so-far fitness attained after 3,000 generations for various mutation rates.

6.3.3 Population Size

This subsection explores the effect of modifying the population size on the magnitude of optimal mutation rates. Four population sizes: 10, 25, 50, and 100, were tested. The number of generations used as a stop criterion varied according to the population size since the smaller the population, the more generations were needed for equilibrating the best-so-far fitness. So the termination criteria

[1]On the other instance (Sento 1) it was not possible to use proportional selection since the fitness function often produced negative values.

used were 30,000, 12,000, 6,000, and 3,000 generations for population sizes 10, 25, 50, and 100 respectively. Figure 6.9 shows results on the two selected problem instances. Optimal mutation rates tended to be smaller, the smaller the population size, this tendency was clearer on Weish 30 (right plot), where optimal mutation rates were $0.5/L$ for a population size of 10, $1.0/L$ for a population size of 20, $1.0/L$ - $1.5/L$ for a population size of 50, and $1.5/L$ for a population of size 100. Notice that for population sizes of 50 and 100, differences in performance for the various mutation rates tend to stabilise.

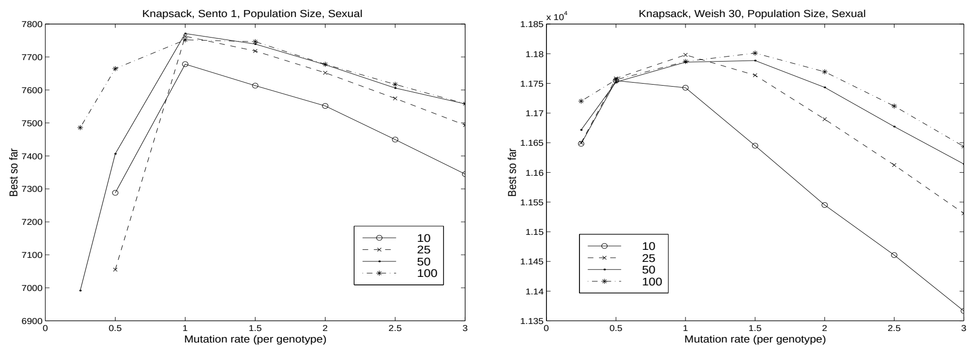


Figure 6.9: Comparing optimal mutation rates for various population sizes (see legends) on two instances of the Multiple Knapsack problem. The curves show the average best-so-far fitness attained after a fixed number of generations for various mutation rates. These fixed number of generations varied according the population size (30,000, 12,000, 6,000 and 3,000 generations for population sizes 10, 25, 50, and 100 respectively).

6.3.4 Elitism

Results on the abstract landscapes explored in Chapter 5 (Section 5.2.5) suggest that elitism has a pronounced effect. When elitism was used, there was no error threshold transition. The following group of experiments explores the effect of including elitism on the magnitude of optimal mutation rates. Before presenting results on the Knapsack instances, the following subsection explores optimal mutation rates with elitism on the same Royal Staircase instance and GA settings for which error thresholds with elitism were investigated in Chapter 5.

Royal Staircase Function

Royal Staircase functions are unimodal and have few fitness values. The single optimum is known beforehand (the string of all 1s). So, a natural performance measure would be to calculate the number of evaluations before reaching the optimum string for the first time. This measure was used by van Nimwegen and Crutchfield (1998) in their study of optimal evolutionary search on Royal Staircase functions. Hence, the performance measure used here on these functions is the number of evaluations before finding the peak, averaged over 100 runs. The landscape instance explored is a Royal Staircase function with number of blocks $N = 3$, and block size $K = 10$. A generational GA with proportional selection is used. The population size is 100. That is, the same settings used on the experiments in Chapter 5 (Section 5.2). Figure 6.10 shows results with and without recombination. Notice that the range of optimal mutation rates is wider for the runs with elitism, also the number of evaluations for finding the peak increases more steadily. On the other hand, the runs without elitism show a sudden increase in the number of evaluations to reach the peak for mutation rates greater than 0.6 (Sexual) and 1.8 (Asexual). The difference between the non-elitist and elitist runs was more marked for the Sexual GA (Figure 6.10, left). Error thresholds without elitism are indicated in the plots. Notice that error thresholds without elitism are located within the range of optimal mutation rates of both the elitist and non-elitist runs.

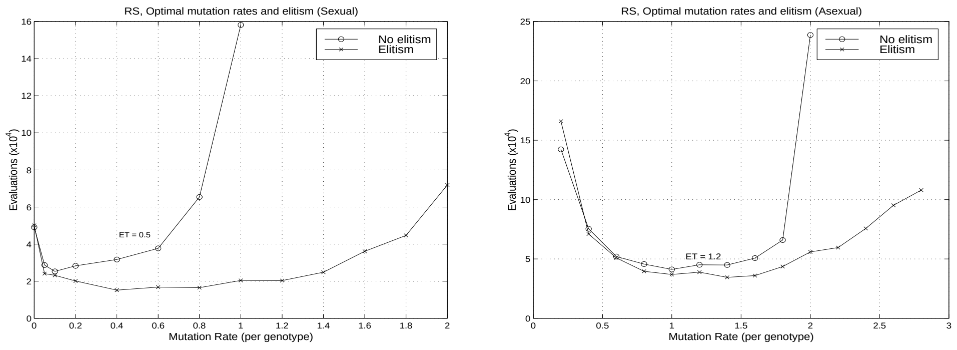


Figure 6.10: Optimal mutation rates and elitism. Number of evaluations for finding the peak on the Royal Staircase function with and without elitism. Left with recombination (Sexual), right without recombination (Asexual). Error thresholds for the non-elitist strategies are indicated in the plots.

Multiple Knapsack Instances

Figure 6.11 compares results with and without elitism on the two selected Knapsack instances. Results suggest that optimal mutation rates are the same with and without elitism. Moreover, the average best-so-far fitness curves are rather similar in both cases. These results differ from those on the Royal Staircase function, which suggest that the effect of elitism is problem dependent.

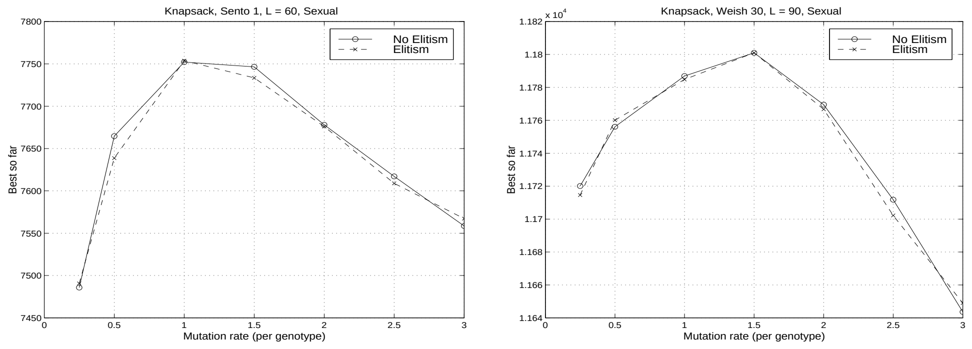


Figure 6.11: Comparing optimal mutation rates for non-elitist and elitist GAs on two instances of the Multiple Knapsack problem. The curves show the average best-so-far fitness attained after 3,000 generations for various mutation rates.

6.3.5 Steady State Population Replacement

The experiments in this subsection explore the effect of using steady-state population replacement instead of generational replacement. Three types of steady-state GA were explored:

1. Using tournament selection for parents, and random selection for individuals that are to be replaced

2. Using random selection for parents, and inverse tournament selection for individuals that are to be replaced

3. Using tournament selection for parents, and inverse tournament selection for individuals that are to be replaced

Figure 6.12 compares results on the Knapsack instances using the three steady-state GAs described above. Since steady-state GAs only replace few individuals (typically one or two) each generation, the number of generations used as the termination criterion needs to be much longer. Specifically, 100,000 generations were long enough to equilibrate the average best-so-far fitness on the Knapsack instances. Results on the two instances are qualitatively very similar. The steady-state replacement of type 1, namely, using tournament selection for parents, produced lower optimal mutation values. On the other hand, the steady-state replacement using inverse tournament selection for individuals that are to be replaced (types 2 and 3), produced higher optimal mutation rates, and a wider range of near-optimal mutation values. The range and magnitude of optimal mutation rates was larger for the steady-state GA of type 3.

The explanation suggested here for this last observation is as follows. The third type of steady-state replacement imposed the highest selection pressure since there was selection on both parents and individuals that are to be replaced. Results with varying selection pressures (Figure 6.8) suggest that optimal mutation rates are higher for higher selection pressures. Thus, steady-state replacement of type 3 imposed the highest selection pressure, and hence produced the highest optimal mutation rates. Regarding the wider ranges of mutation rates observed on the steady-state GAs of types 2 and 3, these results are probably due to their implicit elitism. This is supported by results with elitism on the Royal Staircase function (Section 6.3.4, Figure 6.10).

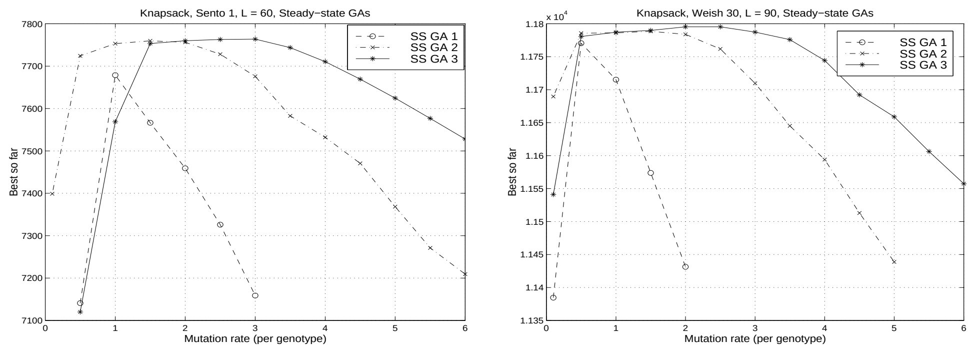


Figure 6.12: Optimal mutation rates and steady-state population replacement. Average best-so-far fitness on two instances of the Multiple Knapsack problem. Three types of steady-state replacement were tested: (1) applying tournament selection on parents and selecting individuals that are to be replaced at random, (2) selecting parents at random and applying inverse tournament selection on individuals that are to be replaced, (3) applying tournament selection on both parents and individuals that are to be replaced.

6.3.6 Discussion

This section explored the effect of modifying the values of various evolutionary parameters on the magnitude of optimal mutation rates. Two instances of the Multiple Knapsack problem were used as test problems. The effects of these various parameters are summarised below:

- **Genotype length:** Optimal per bit mutation rates seem to depend on the string length; they were lower in magnitude the longer the genotype. We consider that expressing mutation rates as mutations per genotype instead of as mutations per bit, is more useful when devising heuristics for optimal setting of the mutation rate.

- **Selection pressure:** The strength of selection had a pronounced effect on optimal mutation rates. The stronger the selection pressure, the higher the magnitude of optimal mutation rates. The use of proportional selection (where there is no control over the selection pressure) may produce much smaller optimal mutation rates as compared to tournament selection. An interesting observation is that for tournament selection with tournament size of 2 (and a population of size 100), optimal mutation rates occurred between 1 and 1.5 mutations per genotype, whereas for tournament size of 4 they increased to 2.5 – 3.0 mutations per genotype (Figure 6.8). This result suggests that selection pressure is the most important component in determining the magnitude of optimal mutation rates.

- **Population size:** The effect of population size on the magnitude of optimal mutation was not found to be marked. However, the evidence suggests that optimal mutation rates are smaller, the smaller the population size. These differences in the magnitude of optimal mutation rates tend to stabilise for population sizes of 50 and larger.

- **Elitism:** Results from Chapter 5 suggest that elitism has a pronounced effect since, when elitism was used, there was no observable error threshold transition. This observation looks problematic from the point of view of the hypothesised relationship between error thresholds and optimal mutation rates. Results on both the Royal Staircase function and Knapsack instances suggest that the relationship between error thresholds (as estimated without elitism) and optimal mutation rates with elitism, is still present since the error threshold is located within the range of optimal mutation rates of both the elitist and non-elitist runs. However, with elitism, the range of optimal mutation rates on the Royal Staircase function was shown to be much wider.

- **Steady State Population Replacement:** Three types of steady-state population replacement were tested. The magnitude and range of optimal mutation rates varied according to the type of steady-state GA, being larger for those types imposing a higher selection pressure. Also, the steady-state GAs with an implicit elitism showed a wider range of optimal mutation rates. However, a mutation rate of 1 – 1.5 mutations per genotype produced near-optimal results in all cases.

6.4 Optimal Mutation Rates and Fitness Landscape Structure

This section explores the effect of modifying the landscape structure on the magnitude and extent of the range of optimal mutation rates. Various parameterisations of the Royal Staircase function are explored. Moreover, results on a group of *NK* and *NKF* landscapes are presented. All the experiments use a generational GA with tournament selection (tournament size of 2), and a population of size 100. Both mutation and recombination are used (Sexual GA). The recombination operator is two-point recombination with a rate of 1.0, that is, a similar setting to that summarised in table 6.3. A wide range of mutation rates were explored, they are expressed as mutations per genotype.

6.4.1 Royal Staircase Functions

Three Royal Staircase functions of fixed string length 32, and different values of *N* and *K* (see Table 6.4), were explored. This produces a range of functions of fixed length and increasing ruggedness. Figure 6.13 shows the average best-so-far fitness attained after a fixed number of generations on each function. The number of generations used as the termination criteria was longer the more rugged the function, as more generations were needed to equilibrate the best-so-far fitness of the population. This number is indicated on each plot title. Notice that (Figure 6.13), when the landscape is smoother (i.e. greater number of steps N and smaller step size K), the range of optimal mutation rates is wider. For the more rugged landscape ($N = 2$, $K = 16$; right plot) only one mutation value ($0.8/L$) produced optimal performance. There is, however, an overlap of optimal mutation rates over these landscapes of increasing ruggedness, with a mutation rate of $0.8/L$ producing optimal performance in all of them.

N	K
16	2
4	8
2	16

Table 6.4: Royal Staircase functions explored.

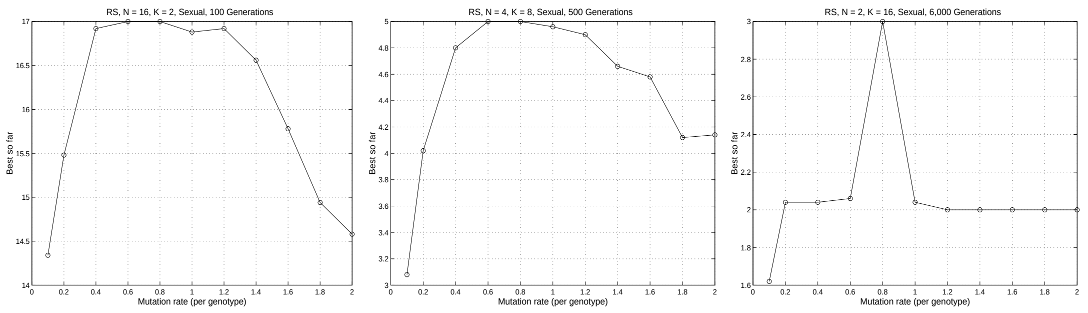


Figure 6.13: Optimal mutation rates on Royal Staircase functions of fixed length and increasing ruggedness (decreasing number of steps). The curves show the average best-so-far fitness attained after a fixed number of generations (shown in the plot titles) for various mutation rates.

6.4.2 *NK* Landscapes

This set of experiments explores optimal mutation rates on NK landscapes of fixed length ($N = 16$), and four values of $K = \{0, 4, 8, 12\}$. This produces a range of landscapes from a single-peaked and smooth 'Fujiyama' landscape ($K = 0$), to a very rugged landscape ($K = 12$). For estimating optimal mutation rates on these landscapes, a different approach was followed. This approach is described below.

Method

Results with the NK landscape were found to strongly depend on the termination criterion. Also, differences in performance for various mutation rates were small and tended to converge for large run times. This is probably due to the random nature and statistical regularity of these landscapes. So, for comparing results the best approach found was to show results over the whole run time of a GA (at fixed intervals), instead of only after a fixed termination criterion. All experiments were averaged over 50 runs. Finally, following the methodological suggestions by Spears (1998) (also used by Smith and Fogarty (1996)) each run uses a different landscape (generated with a different seed). For equivalence, the same 50 seeds and landscapes were used for each algorithm variant under comparison.

Results

Figure 6.14 shows the average best-so-far fitness attained at fixed intervals of the whole run, on the four *NK* landscapes. The *NK* landscape with $K = 0$ is an unrealistic smooth 'Fujiyama' landscape; it is, however, shown here for comparison purposes. In this landscape a high mutation rate ($\geq 4.0/L$) produced poor performance, while the lower mutation rates explored ($\leq 3.0/L$) all produced similar good results. However, after a certain number of generations all performance curves tend to converge. On the landscapes with 'medium' ruggedness (K = 4 and 8), results depend on the stage of the search. However, a rate around 2.0 – 3.0 mutations per genotype can be identified as producing the best performance. On these landscapes the error threshold magnitude, as estimated in Chapter 5, was 1.5 – 2.0 mutations per genotype. Thus, optimal mutation rates were slightly higher than the estimated error threshold on these landscapes. On the very rugged landscape (K = 12) a mutation rate of $2.0/L$ produced the best performance over the whole run. This value is again slightly higher than the estimated error threshold on this landscape (around 1.0/L). These results differ from those on the other problems explored. This difference is probably due to the random nature, high multi-modality, and statistical regularity of *NK* landscapes. Also, for extremely rugged landscapes ($K \simeq N$), a very high mutation rate (close to random search) would probably produce optimal results.

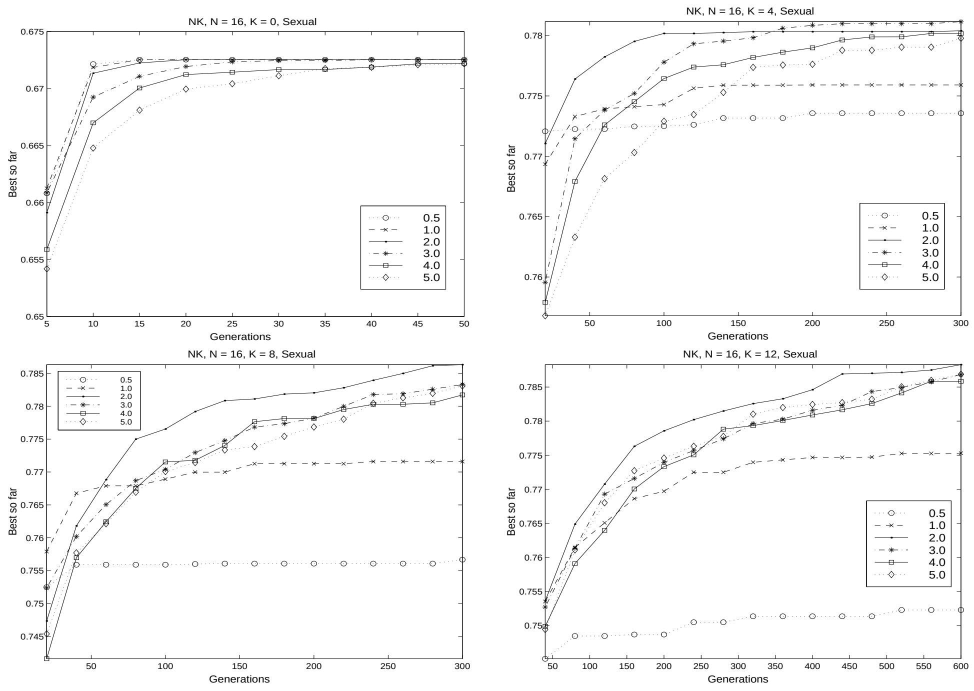


Figure 6.14: Optimal mutation rates on *NK* landscapes of fixed length (N = 16) and increasing ruggedness (K = 0, 4, 8, and 12). The curves show the average best-so-far fitness attained at fixed intervals over the whole run. The legends indicate the mutation rates explored, expressed as mutations per genotype.

6.4.3 *NKF* Landscapes

This subsection explores the effect of landscape neutrality on the magnitude of optimal mutation rates. The empirical approach described above for *NK* landscapes, was used for estimating optimal mutation rates. Figure 6.15 compares results on an *NKF* landscape (Chapter 3, Section 3.2.1) with the maximum possible degree of neutrality ($F = 2$), against those on a standard *NK* landscape of the same ruggedness and dimension. Results suggest that optimal mutation rates depend on the stage of the search, but are higher on the landscape with neutrality as compared to the landscape with no neutrality. Specifically, the optimal mutation rate was $5.0/L$ for the neutral *NKF* landscape as compared to $3.0/L$ for the standard *NK* landscape. These results are consistent with the effect of redundancy in the encoding of the Wing-Box problem (Section 6.2.4).

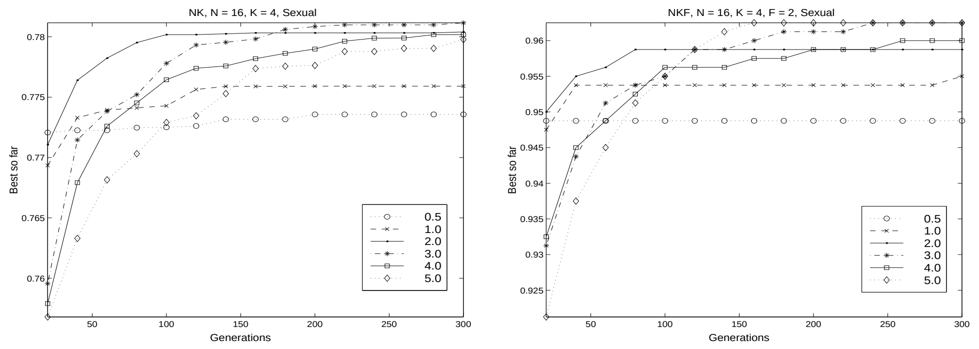


Figure 6.15: Optimal mutation rates on two landscapes of fixed length ($N = 16$) and ruggedness ($K = 4$) and increasing neutrality. The *NKF* landscape (right plot) has the maximum amount of neutrality possible with this model, whereas the *NK* landscape (left plot) has no neutrality at all. The legends indicate the mutation rates explored, expressed as mutations per genotype.

6.4.4 Discussion

This section explored the effect of modifying the landscape structure on the magnitude and range of optimal mutation rates. For the experiments in this section, GA parameters remained fixed, while landscape parameters were varied. On the Royal Staircase function, results suggest that the smoother the landscape, the larger the range of optimal mutation rates and the higher the upper limit. Moreover, GA performance on smooth functions was less sensitive to the particular mutation rate used, in other words, on very rugged landscapes selecting the mutation parameter is more critical from the point of view of the algorithm performance. A similar tendency was observed on *NK* landscapes: smooth landscapes had a wider range of near-optimal mutation rates. Finally, the degree of neutrality or redundancy on the landscape was found to have an effect on the magnitude of optimal mutation rates, which were higher, the higher the degree of neutrality.

6.5 Conclusions

This chapter explored optimal mutation rates over a wide range of both landscape topologies and GA parameter settings. Also, the relationship between error thresholds and optimal mutation rates was assessed by comparing these two measures on both abstract landscapes and real-world domains. It was found that error thresholds and optimal mutation rates are generally correlated.

Moreover, this relationship carried over from simple toy-problems to real-world applications. Also, effects of changing evolutionary parameters on the magnitude of error thresholds, as seen in Chapter 5, occurred in the same proportion on optimal mutation rates found in this chapter, which further confirms the relationship between these two measures.

Optimal mutation rates with and without elitism were found to be similar, although elitism seems to produce a wider range of optimal mutation values. Also optimal mutation rates were independent of the use or not of recombination in most cases. The effect of the size of the population was small, optimal mutation rates are similar for moderate and large populations (more than 50 members). The factors that really determine the magnitude of optimal mutation rates are the strength of selection and the genotype length. We recommend expressing mutation rates as mutations per genotype instead of as mutations per bit. As for the selection scheme, the recommendation is to use rank-based selection methods since they allow the user explicit control of the selection pressure. Moreover, for an otherwise standard GA, using tournament selection with tournament size of two, a mutation rate of 1 – 1.5 mutations per genotype produced good performance in most problem instances studied here. In the presence of neutrality or redundant encodings a slightly higher mutation rate will be optimal. However, since it is difficult to estimate the degree of redundancy beforehand, using a mutation rate of 1 – 1.5 mutations per genotype will be safe and still produce good results. Surprisingly, the ruggedness of the landscape was not critical in determining the magnitude of a near-optimal mutation rate (except perhaps for extremely rugged landscapes). So, the suggestion above holds over landscapes of increasing ruggedness. On a smooth landscape, a wider range of mutation values will produce near-optimal results, this range, however, encompasses the values mentioned above. On the other hand, performance on moderate to rugged landscapes, is more sensitive to an appropriate setting of the mutation parameter.

Chapter 7

Conclusions

The objective of this investigation was to bring the notion of error thresholds from the field of molecular evolution to the field of genetic algorithms, and to establish the relevance of this notion in the context of GAs. More precisely, the aims of this work were the following:

- To establish whether the phenomenon of an error threshold can be observed in populations of bit strings evolving under a GA
- To relate error thresholds to the more familiar notion of optimal mutation rates in GAs
- To propose general principles for setting near-optimal evolutionary parameters in GAs in the light of this new knowledge

To achieve these objectives, empirical methods for estimating error thresholds on landscapes ranging from simple to complex, including real-world domains, were proposed. These approaches were inspired by research from the field of molecular evolution. Thereafter, optimal mutation rates were estimated on the same landscapes, and these two measures were compared against each other to assess their relationship. The effects of modifying both the values of evolutionary parameters and the structure of fitness landscapes on the magnitude of error thresholds were also studied. A similar study was carried out for optimal mutation rates. Finally, some general principles of interaction between mutation rates and other evolutionary parameters were suggested.

7.1 Summary

Chapter 2 introduced the field of evolutionary computation and described in detail the most widely known of its approaches: genetic algorithms (GAs). The different components and variants of GAs were described, revealing the GA as a family of algorithms rather than a single algorithm. To complicate matters further, there is little (if any) theoretical guidance, and few rules of thumb about how to proceed when applying a GA to a given problem. Thus, the chapter also discussed the many decisions involved when designing a GA. Among such decisions, parameter setting was discussed in more detail, and a classification of approaches to parameter setting was proposed. Also, a detailed review of approaches so far for effective setting of the mutation rate was presented.

Chapter 3 introduced the notion of fitness landscapes, which was originally proposed in the context of organic evolution, but later gained relevance in both molecular evolution and evolutionary computation. Landscapes may differ in their structure, hence, some landscape features that are known to have an influence on evolutionary search were discussed. Among such features **ruggedness** and **neutrality** were distinguished. Thereafter, some techniques for analysing the structure of fitness landscapes were briefly discussed. The second part of the chapter presented the test problems used throughout the dissertation. Two types of test problems were selected; first, a group of abstract fitness landscapes (Royal Staircase functions, NK landscapes, and NK landscapes with neutrality); and second, two real-world applications (a combinatorial optimisation problem: the Multiple Knapsack problem, and an engineering application: the design of an optimal aircraft Wing-Box). The families of abstract tunable landscapes allowed the exploration of a wide range of landscape topologies with several degrees of ruggedness and neutrality, whereas the real-world problems allowed us to explore the practical relevance of the ideas in this thesis.

Chapter 4 started the exploration of error thresholds in GAs using simple abstract landscapes. It also introduced the notions of quasispecies and error thresholds from molecular evolution, and discussed the major extensions of the original quasispecies model. Thereafter, it reproduced the results by Boerlijst et al. (1996), but using a GA (and thus finite populations) instead of the quasispecies model (for infinite populations) as the underlying model of evolution. Results for finite populations showed that the stable distribution of sequences was qualitatively similar to that for infinite populations. Thus, error thresholds were shown to exist in finite populations of bit strings evolving under a GA. Moreover, the main conclusions of Boerlijst and co-workers hold in this case; in particular, the main conclusion that recombination shifts the error threshold to lower mutation rates. An additional group of experiments, not included in Boerlijst et al. (1996) were presented. These experiments explored the effect of including mate selection. It was found that assortative mating (i.e. preference for similar organisms) increased the magnitude of error thresholds as compared to no recombination and recombination with random mating.

Chapter 5 introduced the so-called consensus sequence plots. These plots, borrowed and adapted from theoretical biology, represent an empirical approach for locating error thresholds on general landscapes. They also serve as a tool for visualising some features of the landscape structure such as ruggedness and presence of discontinuities. The empirical sections of the chapter used consensus sequence plots for exploring the effect of varying several evolutionary parameters on the magnitude of error thresholds on both abstract landscapes and real-world problems. It was found that the magnitude of error thresholds depends mainly on the strength of selection and the reciprocal of the genotype length. Error thresholds also increase with increasing population size, although these differences in magnitude stabilise for population sizes of 50 or larger. Elitism has a pronounced effect, when elitism was used, no error threshold transition was observed. Error thresholds depended on the type of steady-state replacement used, this difference, however, was attributed to both the differences in the strength of selection and the implicit elitism of some types of steady-state replacement. For discontinuous and very rugged landscapes, error thresholds were found to be lower when recombination was used. However, this effect of recombination was not observed on less rugged landscapes and real-world domains. With regard to assortative mating, error thresholds were higher for this mating scheme as compared to both random mating and no recombination. Regarding the structure of fitness landscapes, it was found that the existence of error thresholds depends upon the ruggedness of the landscape. For smooth landscapes, there was no clear error threshold. For rugged landscapes, on the other hand, there was a clear transition between an 'ordered' (selection-dominated) regime and a 'disordered' (mutation-dominated) one. Finally, empirical evidence suggests that error thresholds also occur in real-world domains.

Chapter 6 explored optimal mutation rates. It discussed a working definition of an 'optimal' mutation rate, and described an approach for estimating optimal mutation rates on the various problems explored. Thereafter, a first group of experiments assessed the relationship between error thresholds (as estimated in Chapter 5) and optimal mutation rates (as estimated in Chapter 6) by comparing these two measures on both abstract landscapes and real-world domains. The empirical evidence gathered suggests a strong correlation between these two measures. However, the optimal mutation rate is often not a well defined value (taking a range), whereas the error threshold is a more definite measure. A second group of experiments explored the effect of modifying the most relevant evolutionary parameters on the magnitude of optimal mutation rates. It was found the most important factors were the strength of selection and the reciprocal of genotype length. Optimal mutation rates were similar with and without recombination on the real-world domains. However, on the abstract problems, they tend to be lower when recombination was used. These results are consistent with the effect of recombination on error thresholds reported in Chapter 5. The effect of the size of the population was also small, optimal mutation rates were similar for population sizes of 50 and larger. The range of near-optimal mutation rates seems to be wider for elitist GAs. The last group of experiments explored the effect of modifying the landscape structure. The ruggedness of landscapes was not critical either, although smooth landscapes showed a wider range of mutation values producing near-optimal performance. Finally, on most problems explored, and for a controlled selection pressure (tournament selection, with tournament size of 2), optimal mutation rates were consistently around 1 – 1.5 mutations per genotype. On redundant encodings, a slightly higher mutation rate would be optimal. However, since the degree of redundancy is not easy to estimate beforehand, a mutation rate of $1/L$ – $1.5/L$ will be safe and still produce good results.

7.2 Contributions

- A classification of approaches to GA parameter setting was proposed. This classification modifies and extends a previous taxonomy by Eiben et al. (1999).

- The notion of error threshold was brought from the field of molecular evolution to the field of genetic algorithms. The existence of the phenomenon of an error threshold in populations of bit strings evolving under a GA was demonstrated over a wide range of landscapes and problems, including real-world domains.

- Consensus sequence plots were also borrowed and adapted from theoretical biology. They serve as an empirical approach for locating error thresholds on general landscapes. Moreover, they represent a new visualisation tool that reveals several features of the landscape structure such as ruggedness and presence of discontinuities.

- It was found that, when comparing the performance of several GA variants, the outcome may depend on the choice of the performance measure. In particular, the choice of the termination criterion, as a fixed number of generations or a fixed number of evaluations (of newly created individuals), may modify the results.

- Most importantly, the hypothesised relationship between the notion of error threshold and the more familiar notion of an optimal mutation rate in GAs was empirically corroborated. Moreover, this relationship was shown to carry over from simple abstract landscapes to real-world domains.

- The effect of modifying the most relevant evolutionary parameters on the magnitude of both error thresholds and optimal mutation rates was investigated. This study revealed several principles concerning the interaction between each of these parameters with the mutation rate, together with the sensitivity of these interactions.

- The heuristic of setting a mutation rate of one mutation per genotype ($1/L$) has been proposed before within the evolutionary computation community. However, results in this dissertation set bounds to the validity of this heuristic. A mutation rate of $1/L$ would be sub-optimal in the following cases:

  - a weak selection pressure,
  - an excessively high selection pressure,
  - a small population size (< 20), and
  - in the presence of highly neutral (redundant) genotypes.

- From the evidence gathered in this dissertation, we suggest that mutation rates should be expressed as mutations per genotype instead of as mutations per bit.

7.3 Limitations

- The number of landscapes and problems explored was necessarily limited. The use of real-world domains as test problems supports the practical relevance of the findings in this dissertation. However, it is worth observing that other real-world problems might have different characteristics from those explored in this dissertation.

- The genetic representation explored was limited to fixed-length binary strings.

- Among the different approaches to evolutionary computation, this dissertation was limited to genetic algorithms.

- The mutation rate was considered in its standard form, that is, as a fixed value throughout the whole GA run.

7.4 Suggestions for Further Study

Since this dissertation focused on fixed-length binary strings, a natural extension would be to explore variable-length, n-ary discrete representations. The extension of the notion of error thresholds to real number encodings, and non-linear (e.g. hierarchical) chromosomes would be a more difficult enterprise. In a similar vein, it would be interesting to explore the existence of error thresholds with other evolutionary algorithms such as evolution strategies, evolutionary programming, and genetic programming. We can expect the phenomenon to persist, as it comes from the quasispecies model, which is a formal model of evolution based on differential equations, different in nature from a computational model such as the GA.

It has been suggested that a mutation parameter that varies across the GA run would be optimal. However, devising optimal schemes for varying the mutation rate is a difficult task. The notion of error threshold might be relevant to this enterprise as it possesses an upper limit to the mutation parameter beyond which evolutionary search would degenerate into random search. In a similar vein, consensus sequence plots may suggest optimal mutation rate schedules. Consensus sequence plots show differences in the error threshold magnitude across the genotype, which are more clearly observed on Royal Staircase functions (Chapter 5, Figures 5.7, 5.17). This supports the idea that a time-varying scheme for the mutation rate would be optimal. This idea was originally proposed by Fogarty (1989), who found that varying the mutation rate over time and across the bit representation of individuals (or both), significantly improved the performance of the GA. Later on, similar findings were reported by Bäck (1992) and Mühlenbein (1992). A clear implication of the findings in this dissertation is that, not only can useful estimates of optimal mutation rates be inferred from error thresholds, but also that a systematic method of setting a non-fixed schedule of such rates can be devised for families of real-world application problems. This, then, deserves further investigation.

Finally, the anomalous results found when estimating optimal mutation rates on GAs without recombination, and the distinction between a termination criterion considering either generations or evaluations of newly created individuals (Chapter 6, Section 6.2.1), suggest a potentially interesting line of research. Specifically, that, at least for landscapes with high levels of neutrality, a mutation-only algorithm with a low mutation rate may produce better performance than a standard GA, when performance is in terms of new evaluations.

7.5 Final Words

The field of genetic algorithms is characterised, in my opinion, by a big gap between the theory and the practice. There are very few theoretical results, and, most of them are not relevant for the practitioner. This thesis is an attempt at bridging this gap. By bringing the notion of error threshold into the field, new light was shed on the sensitivity of the mutation rate parameter. Also, some principles concerning the interactions between the mutation rate and other evolutionary parameters were illuminated. This new understanding is not only relevant from the theoretical point of view, but was used here to reveal potential useful heuristics concerning parameter interactions and near-optimal parameter setting.

Bibliography

Bäck, T. (1991). Self-adaptation in genetic algorithms. In Varela, F. J., & Bourgine, P. (Eds.), *Proceedings of the First European Conference on Artificial Life. Toward a Practice of Autonomous Systems*, pp. 263–271 Cambridge, MA. MIT Press.

Bäck, T., & Schütz, M. (1996). Intelligent mutation rate control in canonical genetic algorithms. In Rás, Z. W., & Michalewicz, M. (Eds.), *Proceedings of the Ninth International Symposium on Foundations of Intelligent Systems*, Vol. 1079 of *LNAI*, pp. 158–167 Berlin. Springer.

Bäck, T. (1992). The interaction of mutation rate, selection, and self-adaption within a genetic algorithm. In und R. Manderick, B. M. (Ed.), *Parallel Problem Solving from Nature II*, pp. 85–94 Amsterdam. Elsevier.

Bäck, T. (1993). Optimal mutation rates in genetic search. In Forrest, S. (Ed.), *Proceedings of the Fifth International Conference on Genetic Algorithms*, pp. 2–8 San Mateo, CA. Morgan Kaufmann.

Bäck, T. (1996). *Evolutionary Algorithms in Theory and Practice*. The Clarendon Press Oxford University Press, New York.

Bäck, T., & Khuri, S. (1994). An evolutionary heuristic for the maximum independent set problem. In *Proceedings of the First IEEE Conference on Evolutionary Computation*, pp. 531–535. IEEE Press.

Baker, J. E. (1985). Adaptive selection methods for genetic algorithms. In Grefenstette, J. J. (Ed.), *Proceedings of the First International Conference on Genetic Algorithms and their Applications*, pp. 101–111 Pittsburgh, PA. Lawrence Erlbaum Associates.

Baker, J. E. (1987). Reducing bias and inefficiency in the selection algorithm. In Grefenstette, J. J. (Ed.), *Proceedings of the Second International Conference on Genetic Algorithms and their Applications*, pp. 14–21 Hillsdale, New Jersey. Lawrence Erlbaum Associates.

Barnett, L. (1997). Tangled Webs: Evolutionary Dynamics on Fitness Landscapes with Neutrality. Master's thesis, School of Cognitive and Computing Sciences, University of Sussex.

Beasley, J. E. (1990). OR-library: Distributing test problems by electronic mail. *Journal of the Operational Research Society*, *41*(11), 1069–1072. Also available at http://www.ms.ic.ac.uk/info.html.

Boerlijst, M. C., Bonhoeffer, S., & Nowak, M. A. (1996). Viral quasi-species and recombination. *Proc. Royal Soc. London B*, *263*, 1577–1584.

Bonhoeffer, S., & Stadler, P. (1993). Error thresholds on correlated fitness landscapes. *Journal of Theoretical Biology*, *164*, 359–372.

Bremerman, H., Rogson, M., & Salaff, S. (1966). Global properties of evolution processes. In *Natural Automata and Useful Simulations*, pp. 3–41. Spartan.

Corne, D., Ross, P., & Fang, H.-L. (1994). Fast practical evolutionary timetabling. *Lecture Notes in Computer Science*, *865*, 250–260.

Davis, L. (1989). Adapting operator probabilities in genetic algorithms. In Schaffer, J. D. (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 61–69 George Mason University. Morgan Kaufmann.

Davis, L. (1991). *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York.

DeJong, K. A. (1975). *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. Ph.D. thesis, University of Michigan, Ann Arbor, MI. Dissertation Abstracts International 36(10), 5140B, University Microfilms Number 76-9381.

Eiben, A. E., Hintering, R., & Michalewicz, Z. (1999). Parameter control in evolutionary algorithms. *IEEE Transations on Evolutionary Computation*, *3*(2), 124–141.

Eigen, M. (1971). Self-organization of matter and the evolution of biological macromolecules. *Naturwissenschaften*, *58*, 465–523.

Eigen, M., McCaskill, J., & Schuster, P. (1988). Molecular quasi-species. *J. Phys. Chem.*, *92*, 6881–6891.

Eigen, M., & Schuster, P. (1979). *The Hypercycle: A Principle of Natural Self-Organization*. Springer-Verlag.

Fogarty, T. C. (1989). Varying the probability of mutation in the genetic algorithm. In Schaffer, J. D. (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 104–109 George Mason University. Morgan Kaufmann.

Fogel, D. (1995). *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence.* IEEE Press.

Fogel, L. J., Owens, A. J., & Walsh, M. J. (1966). *Artificial Intelligence through Simulated Evolution*. Wiley Publishing, New York.

Fontana, W., & Schuster, P. (1987). A computer model of evolutionary optimization. *Biophys. Chem.*, *26*, 123–147.

Futuyama, D. (1998). *Evolutionary Biology* (Third edition). Sinauer Associates, Inc., Sunderland, MA.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, MA.

Goldberg, D. E., & Deb, K. (1991). A comparative analysis of selection schemes used in genetic algorithms. In Rawlins, G. J. E. (Ed.), *Proceedings of the First Workshop on Foundations of Genetic Algorithms*, pp. 69–93 San Mateo, CA. Morgan Kaufmann.

Grefenstette, J. J. (1986). Optimisation of control parameters for genetic algorithms. *IEE Transactions SMC*, *16*(1), 122–128.

Hart, D., & Clark, A. G. (1997). *Principles of Population Genetics* (Third edition). Sinauer Associates, Sunderland, MA.

Harvey, I. (1992). Species adaptation genetic algorithms: The basis for a continuing SAGA. In Varela, F. J., & Bourgine, P. (Eds.), *Proceedings of the First European Conference on Artificial Life. Toward a Practice of Autonomous Systems*, pp. 346–354. MIT Press/Bradford Books, Cambridge, MA.

Harvey, I. (1997). Artificial evolution for real problems. In Gomi, T. (Ed.), *Evolutionary Robotics: From Intelligent Robots to Artificial Life*. AAI Books.

Harvey, I., Husbands, P., Cliff, D., Thompson, A., & Jakobi, N. (1997). Evolutionary robotics: The Sussex approach. *Robotics and Autonomous Systems*, *20*, 205–224.

Harvey, I., & Thompson, A. (1996). Through the labyrinth evolution finds a way: A silicon ridge. In *Proceedings of the First International Conference on Evolvable Systems: From Biology to Hardware*. Springer-Verlag.

Hesser, J., & Männer, R. (1991). Towards an optimal mutation probability for genetic algorithms. In Schwefel, H. P., & Männer, R. (Eds.), *Parallel Problem Solving from Nature*. Springer-Verlag, Lecture Notes in Computer Science Vol. 496.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor.

Holland, J. H. (1986). Escaping brittleness: the possibilities of general purpose algorithms applied to parallel rule-based systems. In Michalski, R. S., Carbonell, J. G., & Mitchell, T. M. (Eds.), *Machine Learning, an Artificial Intelligence Approach*, Vol. 2, pp. 593–623. Morgan Kaufmann, San Mateo, CA.

Hordijk, W. (1997). A measure of landscapes. *Evolutionary Computation*, *4*(4), 335–360.

Huynen, M. (1995). Exploring phenotype space through neutral evolution. Tech. rep. Preprint 95-10-100, Santa Fe Institute.

Julstrom, B. A. (1995). What have you done for me lately? Adapting operator probabilities in a steady-state genetic algorithm. In Eshelman, L. J. (Ed.), *Proceedings of the Sixth International Conference on Genetic Algorithms*, pp. 81–87 San Francisco, CA. Morgan Kaufmann.

Kauffman, S. (1989). Adaptation on rugged fitness landscapes. In Stein, D. (Ed.), *Lectures in the Sciences of Complexity*, pp. 527–618. Addison-Wesley, Reading, MA.

Kauffman, S. (1993). *The Origins of Order: Self-Organization and Selection in Evolution*. Oxford University Press.

Khuri, S., & Bäck, T. (1994). An evolutionary heuristic for the minimum vertex cover problem. In Hopf, J. (Ed.), *Genetic Algorithms within the Framework of Evolutionary Computation*, pp. 86–90 Saarbrücken, Germany. Max-Planck-Institut für Informatik.

Khuri, S., Bäck, T., & Heitkötter, J. (1994). The zero/one multiple knapsack problem and genetic algorithms. In Deaton, E., Oppenheim, D., Urban, J., & Berghel, H. (Eds.), *Proceedings of the 1994 ACM Symposium of Applied Computation*, pp. 188–193. ACM Press.

Kimura, M. (1982). The neutral theory as a basis for understanding the mechanisms of evolution and variation at the molecular level. In *Molecular Evolution, Protein Polymorphism and the Neutral Theory*. Japan Scientific Societies Press and Springer-Verlag.

Kinnear, Jr., K. E. (1997). Genetic programming. In Bäck, T., Fogel, D. B., & Michalewicz, Z. (Eds.), *Handbook of Evolutionary Computation*, pp. B1.5:1–6. Institute of Physics Publishing and Oxford University Press, Bristol, New York.

Koza, J. R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA.

Levenick, J. R. (1991). Inserting introns improves genetic algorithm success rate: Taking a cue from biology. In Belew, R., & Booker, L. (Eds.), *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 123–127 San Mateo, CA. Morgan Kaufman.

Levenick, J. R. (1999). Swappers: Introns promote flexibility, diversity and invention. In Banzhaf, W., Daida, J., Eiben, A. E., Garzon, M. H., Honavar, V., Jakiela, M., & Smith, R. E. (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference*, Vol. 1, pp. 361–368. Morgan Kaufmann.

Lipsitch, M. (1991). Adaptation on rugged landscapes generated by iterated local interactions of neighboring genes. In Belew, R., & Booker, L. (Eds.), *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 128–135 San Diego, CA. Morgan Kaufmann.

Manderick, B., de Weger, M., & Spiessens, P. (1991). The genetic algorithm and the structure of the fitness landscape. In *Proceedings of the Fourth International Conference on Genetic Algorithms"*, pp. 143–150 Los Altos, CA. Morgan Kauffman.

Maynard Smith, J. (1970). Natural selection and the concept of a protein space. *Nature*, *225*, 563–564.

McIlhagga, M., Husbands, P., & Ives, R. (1996). A comparison of search techniques on a wing-box optimisation problem. *Lecture Notes in Computer Science*, *1141*.

Merz, P., & Freisleben, B. (1999). Fitness landscapes and memetic algorithm design. In Corne, D., Dorigo, M., & Glover, F. (Eds.), *New Ideas in Optimization*, pp. 245–260. McGraw-Hill, London.

Mitchell, M. (1996). *An Introduction to Genetic Algorithms*. MIT Press.

Mitchell, M., Forrest, S., & Holland, J. H. (1992). The Royal Road for genetic algorithms: Fitness landscapes and GA performance. In Varela, F. J., & Bourgine, P. (Eds.), *Proceedings of the First European Conference on Artificial Life. Toward a Practice of Autonomous Systems*, pp. 245–254 Paris, France. MIT Press, Cambridge, MA.

Mühlenbein, H. (1992). How genetic algorithms really work: I. mutation and hillclimbing. In Männer, B., & Manderick, R. (Eds.), *Parallel Problem Solving from Nature II*, pp. 15–25. North-Holland.

Newman, M. E., & Engelhardt, R. (1997). Effects of neutral selection the evolution of molecular species. *Proc. Royal Soc. London B*, *225*.

Nowak, M., & Schuster, P. (1989). Error thresholds of replication in finite populations: Mutation frequencies and the onset of Muller's ratchet. *Journal of Theoretical Biology*, *137*, 375–395.

Nowak, M., & Schuster, P. (1992). What is a quasispecies?. *TREE*, *7*, 118–121.

Ochoa, G. (2000). Consensus sequence plots and error thresholds: Tools for visualising the structure of fitness landscapes. In Marc Schoenauer, Kalyanmoy Deb, Günter Rudolph, Xin Yao, Evelyne Lutton, Juan Julian Merelo, H.-P. S. (Ed.), *Parallel Problem Solving from Nature VI*, Vol. 1917 of *LNCS*, pp. 129–138 Paris, France. Springer Verlag.

Ochoa, G., Harvey, I., & Buxton, H. (1999). Error thresholds and their relation to optimal mutation rates. In Floreano, D., Nicoud, J.-D., & Mondada, F. (Eds.), *Proceedings of the Fifth European Conference on Advances in Artificial Life*, Vol. 1674 of *LNAI*, pp. 54–63 Berlin. Springer.

Ochoa, G., Harvey, I., & Buxton, H. (2000). Optimal mutation rates and selection pressure in genetic algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 315–322.

Ochoa, G., & Harvey, I. (1998). Recombination and error thresholds in finite populations. In Banzhaf, W., & Reeves, C. (Eds.), *Foundations of Genetic Algorithms 5*, pp. 245–264. Morgan Kaufmann, San Francisco, CA.

Ochoa, G., Harvey, I., & Buxton, H. (1999). On recombination and optimal mutation rates. In Banzhaf, W., Daida, J., Eiben, A. E., Garzon, M. H., Honavar, V., Jakiela, M., & Smith, R. E. (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference*, Vol. 1, pp. 488–495.

Patterson, C. (1999). *Evolution*. Natural History Museum, London.

Rechenberg, I. (1973). *Evolution Strategy: Optimization of Technical Systems by Means of Biological Evolution*. Fromman-Holzboog, Stuttgart.

Schaffer, J. D., & Eshelman, L. J. (1991). On crossover as an evolutionary viable strategy. In Belew, R. K., & Booker, L. B. (Eds.), *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 61–68 San Mateo, California. Morgan Kaufmann Publishers.

Schaffer, J., Caruana, R., Eshelman, L., & Das, R. (1989). A study of control parameters affecting online performance of genetic algorithms for function optimization. In Schaffer, J. D. (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms* San Mateo CA. Morgan Kaufmann.

Schuster, P. (1994). Extended molecular evolutionary biology: Artificial life bridging the gap between chemestry and biology. *Artificial Life*, *1*, 39–60.

Smith, J. E., & Fogarty, T. C. (1996). Self adaptation of mutation rates in a steady state genetic algorithm. In *Proceedings of the IEEE Conference on Evolutionary Computation*, pp. 318–323 New York. IEEE Press.

Spears, W. M., Jong, K. A. D., Bäck, T., Fogel, D. B., & de Garis, H. (1993). An overview of evolutionary computation. In Brazdil, P. B. (Ed.), *Proceedings of the European Conference on Machine Learning*, Vol. 667 of *LNAI*, pp. 442–459 Vienna, Austria. Springer Verlag.

Spears, W. M. (1998). *The Role of Mutation and Recombination in Evolutionary Algorithms*. Ph.D. thesis, George Mason University, Fairfax, Virginia.

Syswerda, G. (1989). Uniform crossover in genetic algorithms. In Schaffer, J. D. (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 2–9 George Mason University. Morgan Kaufmann.

Tuson, A. (1995). Adapting Operator Probabilities in Genetic Algorithms. Master's thesis, Department of Artificial Intelligence, Edinburgh University.

Tuson, A., & Ross, P. (1998). Adapting operator settings in genetic algorithms. *Evolutionary Computation*, *6*(2), 161–184.

van Nimwegen, E., & Crutchfield, J. P. (1998). Optimizing epochal evolutionary search: Population-size dependent theory. Tech. rep. Preprint 98-06-046, Santa Fe Institute.

Vassilev, V. K., Fogarty, T. C., & Miller, J. F. (2000). Information characteristics and the structure of landscapes. *Evolutionary Computation*, *8*(1), 31–60.

Vassilev, V. K., Miller, J. F., & Fogarty, T. C. (1999). Digital circuit evolution and fitness landscapes. In *1999 Congress on Evolutionary Computation*, pp. 1299–1306 Piscataway, NJ. IEEE Service Center.

Weinberger, E. D. (1990). Correlated and uncorrelated fitness landscapes and how to tell the difference. *Biological Cybernetics*, *63*, 325–336.

Whitley, D. (1989). The GENITOR algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. In Schaffer, J. D. (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 116–123 San Mateo, California. Morgan Kaufmann Publishers, Inc.

Wright, S. (1932). The roles of mutation, inbreeding, crossbreeding and selection in evolution. In Jones, D. F. (Ed.), *Proceedings of the Sixth International Congress on Genetics*, Vol. 1, pp. 356–366.

Wu, A. S., & Lindsay, R. K. (1995). Empirical studies of the genetic algorithm with non-coding segments. *Evolutionary Computation*, *3*(2).