Evolutionary Computation and Experimental Design

Thesis Submitted to the University of Wales for the Degree of

Doctor of Philosophy

By

Meinwen Pryde, B.Eng.
Mechatronics Research Centre
Department of Engineering
University of Wales College, Newport
July 2001

Summary

This thesis describes the investigations undertaken to produce a novel hybrid optimisation technique that combines both global and local searching to produce good solutions quickly. Many evolutionary computation and experimental design methods are considered before genetic algorithms and evolutionary operation are combined to produce novel optimisation algorithms. A novel piece of software is created to run two and three factor evolutionary operation experiments. A range of new hybrid small population genetic algorithms are created that contain evolutionary operation in all generations (static hybrids) or contain evolutionary operation in a controlled number of generations (dynamic hybrids). A large number of empirical tests are carried out to determine the influence of operators and the performance of the hybrids over a range of standard test functions. For very small populations, twenty or less individuals, stochastic universal sampling is demonstrated to be the most suitable method of selection. The performance of very small population evolutionary operation hybrid genetic algorithms is shown to improve with larger generation gaps on simple functions and on more complex functions increasing the generation gap does not deteriorate performance. As a result of the testing carried out for this study a generation gap of 0.7 is recommended as a starting point for empirical searches using small population genetic algorithms and their hybrids. Due to the changing presence of evolutionary operation, the generation gap has less influence on dynamic hybrids compared to the static hybrids. The evolutionary operation, local search element is shown to positively influence the performance of the small population genetic algorithm search. The evolutionary operation element in the hybrid genetic algorithm gives the greatest improvement in performance when present in the middle generations or with a progressively greater presence. A recommendation for the information required to be reported for benchmarking genetic algorithm performance is also presented. This includes processor, platform, software information as well as genetic algorithm parameters such as population size, number of generations, crossover method and selection operators and results of testing on a set of standard test functions.

CONTENTS

Page

Summary.. i

Contents.. ii

List of Tables.. viii

List of Figures.. xiii

Acknowledgements.. xv

Declaration and Statements.. xvi

1. Introduction.. 1-1

1.1. Background.. 1-1

1.2. Aims of Research.. 1-3

1.3. Outline of Thesis.. 1-5

2. Review of Experimental Design and Evolutionary Computation.... 2-1

2.1 Experimental Design.. 2-1

2.1.1 Introduction.. 2-1

2.1.2 Properties of a Good Design.. 2-2

2.1.3 Review of Experimental Design.. 2-3

2.1.3.1 One-at-a-time experimentation.. 2-3

2.1.3.2 Factorial experiments.. 2-4

2.1.3.3 Fractional factorial experiments.. 2-5

2.1.3.4 Randomisation and blocking.. 2-6

2.1.3.5 Simplex method................................ 2-7

2.1.3.6 Analysis of variance........................... 2-9

2.1.3.7 Taguchi method................................. 2-11

2.1.4 Evolutionary Operation.................................. 2-18

2.1.5 Current Experimental Design........................... 2-21

2.2 Evolutionary Computation... 2-22

2.2.1 Introduction.. 2-22

2.2.2 Evolutionary Programming.............................. 2-22

2.2.3 Evolution Strategies...................................... 2-23

2.2.4 Simulated Annealing..................................... 2-24

2.2.5 Genetic Programming.................................... 2-26

2.2.6 Genetic Algorithms.. 2-28

2.3 Summary of the Selection of Methods for Experimentation.... 2-31

3. Evolutionary Operation and Genetic Algorithms......................... 3-1

3.1 Evolutionary Operation... 3-1

3.1.1 Example.. 3-1

3.1.2 EVOP Philosophy.. 3-10

3.1.3 EVOP Modifications...................................... 3-10

3.1.4 Demise of EVOP and Reasons for its Lack of Use... 3-14

3.1.5 Reviews of EVOP... 3-15

3.1.6 Current Use of EVOP..................................... 3-17

3.1.7 AutoEVOP.. 3-18

3.2 Genetic Algorithms.. 3-21

3.2.1 Schemata.. 3-22

3.2.2 Generic Genetic Algorithm.............................. 3-23

3.2.2.1 Initial population.............................. 3-24

3.2.2.2 New population generation.................. 3-24

3.3 Hybrid Genetic Algorithms.. 3-27

3.3.1 Pareto Optimality GAs.................................... 3-28

3.4 Software Selection... 3-31

3.4.1 EVOP Software.. 3-31

3.4.2 GA Software.. 3-32

3.4.3 Criteria for Selection.................................... 3-33

3.5 Summary.. 3-33

4. Small Populations and Hybrids.. 4-1

4.1 Selection of GA Parameters.. 4-1

4.1.1 Population Size.. 4-1

4.1.2 Coding Alphabet.. 4-2

4.1.3 GA and Experimental Design Hybrids.................. 4-2

4.1.4 EVOP and GA Hybrid Method.......................... 4-3

4.1.5 Test Functions.. 4-4

4.2 Selection of GA Operators.. 4-4

4.2.1 Population Selection...................................... 4-4

4.2.2 Crossover Operator...................................... 4-11

4.2.3 Mutation Operator.. 4-11

4.2.4 Generation Gap... 4-12

4.3 Hybrid Testing... 4-13

4.3.1 Initial Migration and Robustness Testing............... 4-13

4.3.2 The Effect of the Generation Gap......................... 4-15

4.3.2.1 Generation gap with a population of twenty 4-16

4.3.2.1.1 De Jong's first function............. 4-16

4.3.2.1.2 De Jong's second function......... 4-21

4.3.2.1.3 Other functions...................... 4-23

4.3.2.2 The influence of the generation gap......... 4-24

4.3.3 The Effect of a Small Number of Generations......... 4-25

4.3.3.1 De Jong's first function........................ 4-25

4.3.3.2 De Jong's second function..................... 4-27

4.3.3.3 Comparison at ten generations............... 4-28

4.3.4 The Effect of Population Size............................ 4-30

4.4 Summary.. 4-31

5. Dynamic Hybrids.. 5-1

5.1 Modification of Micro GAs.. 5-1

5.1.1 Selection of Parameters.................................... 5-1

5.1.2 Control of EVOP element................................. 5-2

5.2 Dynamic Hybrid Testing.. 5-3

5.2.1 Testing on De Jong's First Function..................... 5-4

5.2.1.1 Hybrid One – GEVO-1........................ 5-4

5.2.1.2 Hybrid Two – GEVO-2........................ 5-5

5.2.1.3 Hybrid Three – GEVO-3........................ 5-6

5.2.1.4 Hybrids Four and Five – GEVO-4 and GEVO-5........................ 5-7

5.2.1.5 Hybrids Six and Seven – GEVO-6 and GEVO-7........................ 5-8

5.2.2 Testing on De Jong's Second Function........................ 5-10

5.2.3 Testing on Other Functions........................ 5-15

5.3 Dynamic Hybrid Influences........................ 5-20

5.3.1 Generation Gap........................ 5-20

5.3.2 Effect of EVOP on Search Movement........................ 5-22

5.3.3 Comparison of Dynamic Hybrids........................ 5-23

5.4 Summary........................ 5-25

6. Benchmark Testing and Applications........................ 6-1

6.1. Benchmarking........................ 6-1

6.1.1 Benchmarking for GAs........................ 6-2

6.1.2 Benchmarking for EVOP........................ 6-5

6.2 Testing of GAs........................ 6-6

6.2.1 Selection of Hybrid GAs for Testing........................ 6-6

6.2.2 Benchmark Functions........................ 6-7

6.2.3 Performance of Hybrid GAs........................ 6-9

6.3 Applications of Small Population GAs........................... 6-13

6.3.1 Applications for Hybrid GAs........................... 6-14

6.3.1.1 Example implementation...................... 6-15

6.4 Summary.. 6-18

7. Conclusions and Further Work...................................... 7-1

7.1 Conclusions.. 7-1

7.2 Contributions.. 7-6

7.3 Future Work.. 7-8

8. References.. 8-1

9. Bibliography.. 9-1

LIST OF TABLES

Table Page

2.1. Scheme for one at a time experimentation.................................. 2-4

2.2. An example factorial experiment.. 2-5

2.3. The settings for the eight runs of a 2^{4-1} fractional factorial............... 2-6

2.4. Principles of ANOVA.. 2-9

2.5. Data set... 2-10

2.6. ANOVA table.. 2-10

2.7. L_8 array... 2-15

3.1. Observations and initial calculations for cycle 2.......................... 3-3

3.2. Constants used in EVOP calculations....................................... 3-4

3.3. Calculations for assessing effects.. 3-6

3.4. Calculation of effects for cycle 2.. 3-7

3.5. Observations and initial calculations for cycle 3.......................... 3-7

3.6. Calculation of effects in cycle 3.. 3-9

4.1. Comparison of a sample of results for a population size of five........ 4-10

4.2. Comparison of a sample of selected results for a population size of ten.. 4-10

4.3. Comparison of a sample of selected results for a population size of twenty... 4-10

4.4. Performance of a standard small population GA on De Jong's first function at the 30th generation with a population of 20 individuals. 4-18

4.5. Performance of a hybrid GA with a population of 20 on De Jong's first function at the thirtieth generation............................... 4-18

4.6. Performance of a standard small population GA on De Jong's second function at the thirtieth generation with a population of 20 individuals... 4-21

4.7. Performance of a hybrid GA with a population of 20 on De Jong's second function at the thirtieth generation............................ 4-21

4.8. Results obtained on Schwefel's function.................................. 4-23

4.9. Results obtained on Rastrigin's function.................................. 4-24

4.10. Results obtained on Griewangk's function............................... 4-24

4.11. Performance of a standard small population GA with a population of 20 on De Jong's first function at generation gap 0.7................ 4-26

4.12. Performance of a hybrid GA with a population of 20 on De Jong's first function at a generation gap of 0.7................................ 4-27

4.13. Performance of a standard small population GA with a population of 20 on De Jong's second function at a generation gap of 0.7...... 4-28

4.14. Performance of a hybrid GA with a population of 20 on De Jong's second function at a generation gap of 0.7............................ 4-28

4.15. Performance of a standard small population GA at ten generations.. 4-30

4.16. Performance of a hybrid GA at ten generations........................ 4-30

5.1. Initiation of EVOP in hybrids.. 5-3

5.2. Comparison of GEVO-1, hybrid 1 and standard small population GA with a population of ten and a generation gap of 0.7................. 5-5

5.3. Performance of GEVO2 hybrid GA with a population of ten and a generation gap of 0.7.. 5-5

5.4. Performance of GEVO3 hybrid GA with a population of ten and a generation gap of 0.7.. 5-6

5.5. Performance of GEVO4 hybrid GA with a population of ten and a generation gap of 0.7.. 5-8

5.6. Performance of GEVO6 hybrid GA with a population of ten and a generation gap of 0.7.. 5-9

5.7. Performance of GEVO7 hybrid GA with a population of ten and a generation gap of 0.7.. 5-9

5.8. Performance of PGA with a population of ten and a generation gap of 0.8.. 5-10

5.9. Performance of Hybrid 1 with a population of ten and a generation gap of 0.8.. 5-11

5.10. Performance of GEVO1 hybrid GA with a population of ten and a generation gap of 0.8.. 5-11

5.11. Performance of GEVO2 hybrid GA with a population of ten and a generation gap of 0.8.. 5-12

5.12. Performance of GEVO3 hybrid GA with a population of ten and a generation gap of 0.8.. 5-12

5.13. Performance of GEVO4 hybrid GA with a population of ten and a generation gap of 0.8.. 5-13

5.14. Performance of GEVO5 hybrid GA with a population of ten and a generation gap of 0.8.. 5-13

5.15. Performance of GEVO6 hybrid GA with a population of ten and a generation gap of 0.8.. 5-14

5.16. Performance of GEVO7 hybrid GA with a population of ten and a generation gap of 0.8.. 5-15

5.17. Performance of GAs on Rastrigin's function with a population of ten and a generation gap of 0.8.. 5-17

5.18. Performance of GAs on Schwefel's function with a population of ten and a generation gap of 0.8.. 5-18

5.19. Performance of GAs on Griewangk's function with a population of ten and a generation gap of 0.8.. 5-19

5.20. Performance of GAs on De Jong's first function with a population of ten.. 5-21

5.21. Ranked performance of dynamic hybrids.................................. 5-23

5.22. Rank of overall performance by dynamic hybrids........................ 5-24

6.1. Settings tested on the weighted sphere model.............................. 6-9

6.2. Summary of results obtained on weighted sphere model................ 6-10

6.3. Summary of results obtained on Rastrigin's function.................... 6-11

6.4. Summary of results obtained on Schwefel's function.................... 6-12

6.5. Summary of results obtained on Greiwangk's function................. 6-13

6.6 Path representation.. 6-16

LIST OF FIGURES

Table Page

2.1. The Taguchi loss function.. 2-13

2.2. Order of runs of simple two factor EVOP................................... 2-19

2.3. Typical individual in a GP population.. 2-27

3.1. Five runs of an EVOP cycle... 3-2

3.2. Scheme of letters used in Table 3.3... 3-6

3.3. Rotating square EVOP (ROVOP)... 3-11

3.4. Summary of Simplex EVOP... 3-13

3.5. Welcome screen from AutoEVOP... 3-19

3.6. Entering of factor names... 3-20

3.7. Typical cycle sheet.. 3-21

3.8. The language of genetic algorithms... 3-22

3.9. Three dimensional schema.. 3-23

3.10. Generic Genetic Algorithm.. 3-23

4.1. Test Functions... 4-5

4.2. Sample comparison of Stochastic Universal Sampling and Roulette Wheel Selection for a micro-GA with a population of five.......... 4-8

4.3. Sample comparison of Stochastic Universal Sampling and Roulette Wheel Selection for a micro-GA with a population of ten........... 4-8

4.4. Sample comparison of Stochastic Universal Sampling and Roulette Wheel Selection for a micro-GA with a population of twenty........ 4-9

4.5. The effect of migration rate on the solution found........................ 4-14

4.6. The effect of generation gap on the solutions found..................... 4-15

6.1 Positions of the cities.. 6-16

ACKNOWLEDGEMENTS

I would like to thank the many people who helped me during these studies, especially my parents for their constant support. My thanks to Hefin, for our discussions and his supervision of this study, without whom this thesis would not have been possible. Thank you to Matthew for all the encouragement to keep going. Finally I must also thank my colleagues in the Department of Engineering for their support and discussions over many cups of coffee.

Diolch yn fawr i bawb.

DECLARATION

This work has not previously been accepted in substance for any degree and is not concurrently submitted in candidature for any degree.

Signed..........Pryde..........(candidate)

Date..........31/7/01..........

STATEMENT 1

This thesis is the result of my own investigations, except where otherwise stated.

Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed..........Pryde..........(candidate)

Date..........31/7/01..........

STATEMENT 2

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed..........Pryde..........(candidate)

Date..........31/7/01..........

1. INTRODUCTION

The aim of this chapter is to illustrate the general background to the areas studied, the aims of the research and to give an outline to the structure of the thesis.

1.1 BACKGROUND

Optimisation is a mature area of research which could be argued to be as old as life itself, as optimal performance gives the greatest chance of survival. Optimisation became more formalised with the development of mathematics, yet some of the more complex real life problems remain impossible to solve using mathematics. Research continues to push the boundaries of mathematics, but other research fields use a different approach by applying methods from other subject areas, such as evolutionary computation.

Experimental design is a more recent subject of primarily the last hundred years. Firmly grounded in mathematics, particularly statistics, experimental design was initially a response to the practical problems of agriculture and industrial chemistry, such as optimising crop or process yield. The requirement was to determine a method to discover the influence of sometimes uncontrollable parameters. Experimental design has developed beyond its original fields of application and has seen a renaissance of interest fuelled by the quality movement in industry. There are many different methods of experimental design, but all attempt to discover the influence of parameters to allow for their control and hence the optimisation of a process.

Evolutionary computation is often perceived as new area of research, but its roots stem from the 1940's. This perception is probably due to the small amount of research that continued after a paper published in the late 1960's by Minsky and Papert (1969) which painted a bleak future for artificial intelligence. The past decade has seen a tremendous growth in research and applications of evolutionary computation. The field encompasses many techniques including genetic algorithms, neural networks, evolutionary strategies and simulated annealing.

Evolutionary operation (EVOP) is a relatively simple experimental design technique, originally developed for application in the chemical industry in the 1950's. EVOP is a little used technique, possibly due to the large number of calculations required, to be quickly completed for the originally intended flow line process applications. EVOP was published decades before the advent of widely available, low cost, computational power. EVOP is considered as a local search method as investigations are carried only a small distance from any current point in a search space.

Genetic Algorithms (GAs) trace their history to the 1970's to the work of Holland (1992) and are based on Darwins' survival of the fittest theory. GAs are also an optimisation technique and as the name implies are based on the behaviour of genes in the natural world. GAs are a global search method that work in a parallel manner, with potentially many points being investigated simultaneously over a large area of any search space. GAs have been applied to large space optimisation problems, such as designing VLSI layouts (Schnecke, 1997) and scheduling (Yamada and Nakano, 1995).

Combining a local search and global search technique has the potential to develop an improved search method that is capable of both types of search. It is in this area of overlap that the studies for this thesis are carried out. By combining EVOP and GAs it is possible to study the influences on performance and potentially develop an improved search technique.

1.2 AIMS OF RESEARCH

The aim of this thesis is to investigate the areas of evolutionary computation and experimental design to determine specific methods that could be combined into a novel approach for optimisation. The hybrid technique should be a robust method which can find good, but not necessarily the absolute best, solutions in a comparatively short length of time, which would increase the potential application areas of GAs from mainly off-line situations, such as scheduling, to other areas such as on-line optimisation and machine control.

Therefore the objectives of this thesis are to:

- Investigate evolutionary computation and experimental design to select methods from each domain to use as a hybrid optimiser with improved performance.
- Develop a deeper understanding of the selected methods.
- Review available software.
- Develop a new piece of software to implement two or three factor EVOP.
- Review the literature to investigate work related to small population GAs.
- Investigate by experiment the most suitable method of selection for small population GAs.
- Investigate by empirical testing, controllable parameters such as generation gap, the size of the population and the number of generations, that influence the performance of a combined hybrid optimiser to establish the best settings for the hybrid technique.
- Investigate the influence of the EVOP operator to determine if including the operator at different generations influences the quality of solution found.
- Determine the level of performance of the hybrid optimisation technique by benchmarking.
- Investigate the difficulties in establishing the exact settings used by authors of published works.
- Examine the elements that should be included in a GA benchmark and its potential to establish precisely how operators influence GA performance.
- Consider applications suitable for small population hybrid GAs.

1.3 OUTLINE OF THESIS

The next chapter, chapter two; Review of Experimental Design and Evolutionary Computation, is an overview of the fields of experimental design and evolutionary computation. The first area to be examined is experimental design: The brief introduction concentrates on the main directions of research in this field, before examining the properties required for a good experimental design. The fundamental experimental designs and some important features are discussed, before progressing to some of the more fashionable methods. Although these techniques have recently risen in prominence they are based on the same principles and often the same designs as those discussed earlier. The Taguchi method is examined in detail as it has been offered as a solution to many problems, although not often in conjunction with evolutionary computation, and offered a potential way forward for these studies. Evolutionary Operation is the final experimental design technique to be examined before a review of the current use of experimental design.

The next section of chapter two considers evolutionary computation and briefly describes the main techniques available, before examining the properties of all the techniques and selecting two methods to be examined in more detail.

Chapter three, Evolutionary Operation and Genetic Algorithms, is a more detailed examination of the two methods selected in chapter two, namely evolutionary operation (EVOP) and genetic algorithms (GAs), the two methods later used to produce new hybrid methods. Details of applications of these methods are given and the availability of software discussed. AutoEVOP software created during the studies for this thesis is then described. A review of genetic algorithm hybrids is undertaken before describing the selection of software for the hybrid methods tested in this thesis.

Chapter four, Small Populations and Hybrids, illustrates the processes undertaken to select suitable operators for the genetic algorithms used in this study and to show any influences these operators have on hybrid GAs in comparison to standard small population GAs on a set of test functions. Initial study is devoted to selecting appropriate parameter values for the GAs, such as the coding alphabet and the size of populations to be examined. Further consideration is then given to the classic operators such as selection method, crossover and mutation for these small population GAs. A range of sets of experiments are undertaken to determine the influence of these operators, especially population size and generation gap.

Chapter five, Dynamic Hybrids, illustrates the investigations undertaken to examine the effects of the influence of the EVOP element on the hybrid GAs, using various standard test functions. Hybrids tested in the previous chapter contained an element of EVOP which operated from the first to last generations of the GA and are henceforth referred to as static hybrids. In this chapter EVOP is initiated only for certain specified generations. As the GA proceeds its characteristics change and these GAs are referred to as dynamic hybrids. These dynamic hybrids are compared to each other, to their comparable static hybrid and standard small population GA. A study is also undertaken into the influence of the generation gap on these dynamic hybrids.

Chapter six, Benchmark Testing and Application, discusses the benefits and problems of benchmarking and its applicability to GA and GA hybrid research. The current use of benchmarking for GAs is then examined, before an illustration of the investigations undertaken to examine the performance of the hybrid GAs on a range of further 'benchmark' test functions. The areas of application for small population GAs are discussed before considering suitable applications for the hybrid GAs studied in this thesis, illustrated with an example implementation.

Finally chapter seven, Conclusions and Further Work, summarises the studies carried out, highlighting the contributions made by this work before discussing how these studies could be extended. Additionally the future possibilities for GAs, EVOP and benchmarking are considered.

2. REVIEW OF EXPERIMENTAL DESIGN AND EVOLUTIONARY COMPUTING

The aim of this chapter is to give a brief overview of experimental design and evolutionary computation. Some of the methods included in these areas are described with an emphasis placed on evolutionary operation and genetic algorithms, the main techniques utilised in this thesis.

2.1 EXPERIMENTAL DESIGN

2.1.1 Introduction

There is always room for improvement. Often to gain the knowledge necessary to improve an industrial process an experiment is conducted, this is the role of experimental design. Inactive observation of a process does not yield sufficient information for improvement, as G.E.P. Box is often quoted "To find out what happens to a system you have to interfere with it (not just passively observe it)" (Box, 1957). The principles of experimental design are based on the work of Sir Ronald A. Fisher in the 1920's and 1930's (Fisher (1925) and Fisher (1935)). Recently interest in experimental design has been revitalised by the work of Genichi Taguchi (1987), who has brought the design of experiments to a wider field of engineering.

Experimental design, once thought to be the preserve of laboratory scientists, transferred a long time ago to the industries of agriculture and chemistry but did not make significant contributions in other business sectors. Although experimental design is a well established field it was generally overlooked by Western manufacturers until the quality of products from the Far East improved dramatically without the expected large price increase; this galvanised Western industry into taking action to remain competitive. It was found that statistical techniques, including experimental design, had aided this improvement. Ironically it was the ideas of Western statisticians, such as Deming (Deming, 1950, 1995), that countries such as Japan embraced after the Second World War, which the West had ignored, that were part of the success story. The techniques have been further developed by Taguchi (1987) into a cohesive system for improving products from the design stage onwards, with the aim of complete customer satisfaction.

Quality of products became increasingly important as availability became less of an issue. The traditional methods of post-production inspection were not competitive enough. Experimental design gave the advantage that many configurations of influential parameters could be tested in a systematic way and the best selected for production.

2.1.2 Properties of Good Designs

All good experiments are pre-planned and implementation has a well-defined structure. Planning is a critical part of any experiment and it is often neglected, (Coleman and Montgomery, 1993). The first stage is to state the purpose of the experiment in clear terms; for a worthwhile experiment to be carried out, there needs to be a clear, definable objective. The next step is to choose which factors, elements of the process such as temperature or machine to be used, are to be studied and at what levels or settings. A response variable, or set of variables, that can be accurately measured must also be selected. If careful thought is given to the preceding stages then selection of a suitable standard experimental design should be comparatively easy. With planning complete the experiment can be run then the data analysed. For a well designed experiment the data analysis involves relatively simple statistical techniques. However, no amount of elegant statistics can rescue a badly planned experiment. With the use of computer programs, analysis can be simple and fast. Graphs are potentially one of the most useful tools at this stage as they allow quick visual interpretation of results. The final stage is to reach conclusions about the experiment and make recommendations for possible changes in the process and/or further experimentation.

2.1.3 Review of Experimental Design

2.1.3.1 One-at-a-time experimentation

The most basic type of experimental design is where one factor at a time is varied, e.g. in a chemical reaction initially all factors are held constant and the yield noted. All factors are held constant except the temperature which for the second experiment is lowered and for the third run the raised, the yield is noted at all settings. All factors are then reset to the original levels whilst only, say, the pressure is decreased then increased. This is shown in Table 2.1.

TEMPERATURE	PRESSURE	YIELD
NORMAL	NORMAL	*Reading 1*
LOW	NORMAL	*Reading 2*
HIGH	NORMAL	*Reading 3*
NORMAL	LOW	*Reading 4*
NORMAL	HIGH	*Reading 5*

Table 2.1. Scheme for one at a time experimentation.

The experiment will give the effect on the yield of a change in temperature or pressure, but will give no indication if increasing both together gives a better yield, i.e. it does not give any interaction effects of the factors and there is no way of discerning if any changes in yield are due to the different factor levels or experimental error. The problem of designating the cause of changes can be overcome by repeating the experiment so that each point in the design space has at least two readings. This allows for the estimation of errors.

2.1.3.2 Factorial experiments

Factorial experiments offer an improvement on 'one at a time' experiments as they incorporate the effect of interaction into the design. Here, using the previous example, trials would be run at low pressure and low temperature (A); low pressure and high temperature (B); high pressure and low temperature (C); high pressure and high temperature (D) as shown in Table 2.2

below. This is a two factor (pressure, temperature), two level (low, high) experiment which has 2^2 (4) runs. This method is acceptable for low order experiments but it soon becomes cumbersome with a greater number of factors and levels, e.g. a three factor, five level experiment would require 3^5 (243) runs. As the size of the experimental space increases, the cost of running the trials will increase in terms of time, money and other resources. It may be that the cost of running the experiment becomes prohibitively expensive and impractical.

	PRESSURE LOW	PRESSURE HIGH
TEMPERATURE LOW	(A)	(C)
TEMPERATURE HIGH	(B)	(D)

Table 2.2. An example factorial experiment.

2.1.3.3 Fractional factorial experiments

Fractional factorials were developed to overcome the problems associated with large experimental spaces. They do not give a blanket coverage of the space but give a good indication of areas where the yield may increase, with a fraction of the runs required by a full factorial experiment. Fractional factorials do however require a more rigorous mathematical approach for their analysis.

Notation for fractional factorials: N^{k-p}

where, N = number of levels

k = number of variables

p = fractional reduction

A 2^4 full fractional factorial design requires sixteen trials but a half fraction requires only eight. $[(½ \times 2^4) = (2^{-1} \times 2^4) = 2^{4-1} = 2^3 = 8]$. Table 2.3 shows the settings for a 2^{4-1} fractional factorial experiment using the traditional designations of '+' for the higher factor level and '-' for the lower level.

TRIAL	A	B	C	D
1	-	-	-	-
2	+	-	-	+
3	-	+	-	+
4	+	+	-	-
5	-	-	+	+
6	+	-	+	-
7	-	+	+	-
8	+	+	+	+

Table 2.3. The settings for the eight runs of a 2^{4-1} fractional factorial.

2.1.3.4 Randomisation and blocking

According to statistical practice the order of the runs should be planned so that it does not influence the test results. To eliminate the problem of environmental factors, such as machine warm up, machine operator etc. randomisation is introduced as protection against the inaccuracies that can occur due to these nuisance variables. Randomisation can be restricted to suit the experiment and there are randomisation tables available to assist the choice of trial order. In practice it is often not possible to randomise an experiment due to set up times and associated increases in costs. However Taguchi (1987), see also section 2.1.3.7, does not consider the randomisation of runs to be important. If it is costly to change factor levels the Taguchi method advocates that in this situation it is best not to randomise the trials as this can lead to erroneous results.

Blocking is another method of reducing experimental error which '...can eliminate the effects of extraneous variations and inhomogeneities in the experimental material and the environment and, hence, improve the efficiency of ...' the designs (Bisgaard, 1994). If there is a large, unavoidable variation from one end of the experimental space to the other then this method, first used by Fisher (1925) for agricultural experiments, is available to reduce the errors this causes. The experimental units are divided into small blocks by a factor, background or nuisance variable, such as time, location of trial, machine, operator or batch of raw materials used. Randomised block design is used when there are more than two factors and one background variable for defining the blocks. Each factor level must occur an equal number of times in each block, if there are only two factor levels then this type of design is called a paired comparison experiment. However, factor levels do not have to occur an equal number of times in each block in an incomplete block design. Blocking also has the advantage that if a mistake is made then only the block of trials which contains the mistake needs to be re-run, not the whole experiment. Randomisation and blocking are important concepts in traditional experimental design and descriptions are to be found in most in experimental design texts, such as Montgomery (1991), Hicks (1982) and Box et al (1978).

2.1.3.5 Simplex method

The simplex method is a response surface method used for the study of mixtures. Simplex progresses to the optimum by moving away from the worst recorded point of the triangular experimental region. It is different from factorial designs in that it is the proportions of the mixture under study that are important and not the amount. The total amount of mixture should remain constant. Introductions to the simplex method can be found in Cornell (1990) and Montgomery (1991). Much of the early work was carried out by Scheffé (1958) who introduced the simplex lattice designs that are still in use today, where the points of experimentation are evenly spread over the design space. A regular simplex design has all sides of equal length, to support a model of degree m in q components over the $\{q,m\}$ lattice. The points of experimentation should be evenly spaced at $(m+1)$ points for values of zero to one for each component proportion.

The points of experimentation do not have to be restricted to those recommended by the lattice design. The two main strategies for choosing experimental points are the distance based strategy and using optimal design criterion. The lattice design is an example of the distance based strategy where the vertices are used for some of the points and the rest are spread evenly across the design space. An example of optimal design is D-optimality which selects points in the design space so that in the response surface model the variance of the regression coefficients are minimised. Experimentation does not have to cover the entire design region, it can be restricted by design or through practical constraints. Simplex was designed for experimentation with mixtures and as such is excellent within this domain, e.g. the determination of the formulation of a household product (Heinsman and Montgomery, 1995) and vinyl car seat covers (Nachtshiem et al, 1990). The best designs are always domain specific, incorporating knowledge of that area.

2.1.3.6 Analysis of variance

Analysis of data is always needed no matter which experimental method is used. One of the most common methods used is analysis of variance (ANOVA). It is a technique for estimating how much of the total variation in a set of data can be attributed to one or more assignable causes of variation, the remainder, not attributable to any assignable cause, being classed as the residual or error variation. It also provides a test of significance and gives a measure of confidence in the data results. The principles of ANOVA are expressed in Table 2.4.

Source of variation	**Degree of freedom**	**Sum of squares***	**Mean square and quantity estimated**
Between means of batches	$(b - 1)$	$k\, n\, \Sigma(y_b - y)^2$	$(MS)_2$ ---$\sigma_0^2 + n\sigma_1^2 + kn\sigma_2^2$
Between samples within batches	$b (k - 1)$	$n\, \Sigma(y - y_b)^2$	$(MS)_1$ --- $\sigma_0^2 + n\sigma_1^2$
Analytical Error	$b\, k (n - 1)$	$\Sigma(y - y_s)^2$	$(MS)_0$ --- σ_0^2
Total	$b\, k\, n - 1$	$\Sigma(y - y)^2$	

KEY	b= batches	y = trial data	σ_0^2 = variance of analytical error
	k = samples	MS = mean square	σ_1^2 = variance of sampling
	n = no. of repetitions		σ_2^2 = variance between batches

* In practice it is more convenient to calculate the sum of squares from the sample and the batch totals (T_s, T_b), e.g. sum of squares between batches = $\Sigma(T_b - T) / kn$,
sum of squares between samples = $\Sigma(T_s - T_b) / n$

Table 2.4. Principles of ANOVA.

Degrees of freedom gives a measure of the amount of information that can be determined from a set of data. For example, if a factor has three levels then the first level can be compared with the other two levels but not itself, so it would have two degrees of freedom. The sum of squares is a measure of the deviation of the experimental data from the mean value of the data, summing each squared deviation emphasises the total deviation. Variance measures the distribution of the data about the mean of the data.

A mean square is a sum of squares divided by its degree of freedom, which gives an unbiased estimate of a population variance.

A simple example of ANOVA (Hicks, 1982), is given below in Table 2.5 and Table 2.6. The aim is to determine if there is a difference in weight loss between materials when under controlled use. Let the factors studied be A, B, C and D and the readings are weight loss in grams.

FACTOR	A	B	C	D
RUN 1	1.93	2.55	2.40	2.33
RUN 2	2.38	2.72	2.68	2.40
RUN 3	2.20	2.75	2.31	2.28
RUN 4	2.25	2.70	2.28	2.25

Table 2.5. Data set.

Source of variation	Degrees of freedom	Sum of squares	Mean square
Between ABCD	(4 -1) = 3	0.5201	0.1734
Error	(4 (4 - 1)) = 12	0.2438	0.0203
Total	(3 + 12) = 15	0.7639	

Table 2.6. ANOVA table.

With this analysis complete further investigation of the results can be carried out, such as the F-test or Scheffé test etc. For the above example, using the F-test;

$$F_{3,12} = \frac{0.1734}{0.0203} = 8.53 \qquad \text{Equation 2.1}$$

using statistical tables, widely available in most statistics text books, the hypothesis that there is considerable difference between average wear resistance of the materials is rejected.

2.1.3.7 Taguchi method

Much of the recent explosion of interest in quality topics, particularly in manufacturing, can be attributed to the work of Genichi Taguchi (1987). His work is based on the quality control principles promoted by Western statisticians such as Deming (1950 and 1960). Quality was taken one step forward by placing it further back in the production cycle meaning that quality could be monitored during manufacture and not as traditionally done after production, allowing more timely intervention if necessary. These principles provided the foundation for the work of Taguchi, who placed quality earlier in the product life cycle, at the design stage. Quality is now often routinely considered from the inception of the design through all manufacturing stages to the customer.

The philosophy of Taguchi can be expressed as: after Roy (1990)

- Quality should be designed into a product not inspected in;

  from design to customer, summarised as "Right first time, right every time".

- There should be minimum deviation from the target value;

  the further from the target value the greater the loss and the worse the performance, which is expressed mathematically by the quadratic loss function

- The cost of quality should be considered system wide and in monetary terms;

  if a product is not of target quality then there are system wide losses to be considered including the cost of scrap, re-work, warranty repairs and others such as the cost in terms of market share loss due to customer dissatisfaction.

To achieve the desired quality Taguchi developed his method which consists of three stages:

• System design.	Choice of factors to be studied and their working levels.
• Parameter design.	Selects the best factor levels.
• Tolerance design.	Fine tuning of factor levels to their optimum.

Engineers tend to think of tolerances and targets in terms of technical specifications such as metres, Ångstroms, kg/m, but management think in monetary terms. Taguchi links the two by expressing quality in financial terms. If a product's quality characteristic is not on target then there is an associated loss. This is best described with the aid of a diagram, see Figure 2.1. It is quite probable that the quadratic loss function is not a precise description of the loss but it has been demonstrated to be adequate, especially since it is difficult to find the data for more complex, accurate models. The mathematical expression used to describe the loss function is:

for a single product $L(Y) = k\,(Y - Y_o)^2$ Equation 2.2

for multiple samples $L(Y) = k\,(MSD)$ Equation 2.3

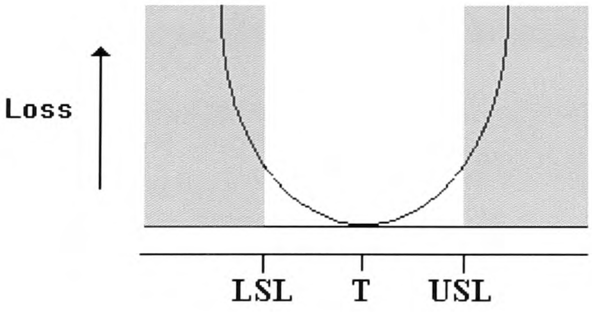
where Y = observed quality characteristic

Y_o = target value for quality characteristic

k = constant

MSD = mean squared deviation of the quality characteristic from the target value

MSD varies depending on whether a specific target value, the smallest or largest possible value is the optimum.



KEY				
	▨	Traditional unacceptable quality	LSL	lower specification limit
	--	Loss function	USL	upper specification limit
	T	Taguchi target specification		

Figure 2.1. The Taguchi loss function.

Starting a Taguchi experiment usually involves a brainstorming session. At this stage all aspects of the experiment should be discussed by a team of people from all the departments involved throughout the life of the product, e.g. from design, production and marketing, and a chairperson. At least one team member should be trained in Taguchi methods and the meeting held, preferably, on neutral ground. The discussion brings in process knowledge at an early stage of the experiment and a multi-disciplinary team is established.

Once it has been decided which factors are to be studied and in which range, an experimental design can be selected. There are an almost limitless number of factorial designs but Taguchi defined a set of useful fractional factorials, called orthogonal arrays, and standardised a method of analysis. Hence Taguchi experiments have a reproducibility, Taguchi has "...simplified and standardised the fractional factorial designs in such a manner that two engineers conducting tests thousands of miles apart, will always use similar designs and tend to obtain similar results." (Roy, 1990). These designs are often wrongly attributed to Taguchi himself; most of them are traditional designs, e.g. L_{16} is a 2^{15-11} due to Finney and L_{32} is due to Fisher, for more examples see Box et al (1988). If the chosen number of factors and levels do not fit into a standard array then the design can be modified to accommodate them, e.g. see Shoemaker and Kacker (1988). Orthogonality means that for each pair of columns all combination of factor levels occur an equal number of times. An example of an orthogonal array is L_8, as shown in Table 2.7.

Note that Taguchi uses ‘1’ to represent the lower level of the factor and ‘2’ to represent the factor at the higher level, compared to the traditional use of ‘-’ and ‘+’.

TRIAL	**FACTORS**						
	A	***B***	***C***	***D***	***E***	***F***	***G***
1	1	1	1	1	1	1	1
2	1	1	1	2	2	2	2
3	1	2	2	1	1	2	2
4	1	2	2	2	2	1	1
5	2	1	2	1	2	1	2
6	2	1	2	2	1	2	1
7	2	2	1	1	2	2	1
8	2	2	1	2	1	1	2

Table 2.7. L_8 array.

The factors should be assigned to suitable columns. Orthogonal arrays work well when there is minimum interaction between factors, i.e. the factors are independent and have a linear effect.

Noise factors need to be identified and included in the experiment. This is done with outer arrays. If there is a large amount of noise present then repetition can help the elimination of its effects. As with all other experimental designs the principles of randomisation apply. Once complete, analysis of the results is needed. Here Taguchi allows the use of standard analysis such as main effect analysis and ANOVA if a standard design has been used, with no repetitions and no interaction effects. For other situations Taguchi has developed signal to noise ratio analysis. This ratio, measured in decibels, is given by:

$\text{S/N Ratio} = -\ 10 \log_{10} (\text{MSD})$ Equation 2.4

and can be used for all situations as the calculation of MSD (mean squared deviation of the quality characteristic from the target value) depends on the desired outcome. The aim is always to find the largest signal to noise ratio (S/N).

Taguchi experimenters often use ANOVA to study the influential factors and plots of marginal means to use a 'pick the winner' approach. This method requires the input of engineering knowledge before optimum settings are selected. If the settings were not part of the original experiment then a confirmation run is carried out.

The Taguchi method is a very useful tool, but it is not a panacea and it does have limitations. The method assumes that there is *a priori* knowledge about interactions, that they are linear and there are no interactions between design and noise factors. Taguchi's reasoning is that if the main effects are controlled, then interactions resulting from those main effects will also be controlled. Problems can arise when unknown interactions are present which can lead experimenters to inaccurate conclusions (Hurley, 1994). Although brainstorming is a very positive aspect of this method it leads to a large single experiment, there is no build up of knowledge as with smaller sequential experiments. Linear graphs used to assign factors to columns can lead to inefficient designs as the aliasing is not clear (McGovern 1994a and McGovern 1994b). An alias occurs when an effect cannot be distinguished from another effect in the design. Taguchi's 'pick the winner' can fail to find the optimum solution even with a full factorial. The signal to noise ratio is an important concept in Taguchi methods but has been widely criticised, e.g. Montgomery (1991), Box (1988), Pignatiello and Ramberg (1991), Tribus and Szonyi (1989), as being inefficient and complex. Classical experimental design often attains the same or better results more clearly (Lucas, 1994). Simpler treatment of the data is recommended such as separation of the mean and standard deviation and use of data analytic methods (Vining and Myers, 1990). The Taguchi method is not statistically advanced and has been criticised as such by statisticians but it is practical and as Pignatiello and Ramberg (1991) point out ".... a method that is understood by a team may be a better choice than one that is slightly more statistically efficient yet only understood by a few." Also as a defence and explanation of Taguchi methods:

> "They are based on the design of experiments to provide near optimal quality characteristics for a specific objective. They are often demeaned by academia for technical deficiencies which are improved by using response surface methodology. Unfortunately most of those who demean Taguchi methods have missed the whole point. Taguchi methods are not a statistical application of designs of experiments. Taguchi methods include the integration of statistical design of experiments into a powerful engineering process."
>
> (Unal and Dean, 1995).

Positive aspects of the Taguchi method include the brainstorming session as it brings together a team of people and pools their knowledge. Teamwork and involvement of all levels of employee is a critical factor in many of the successful experimental design methods. The gathering of as much relevant information before experimentation reduces the problem that the best time to design an experiment is when it is complete. Perhaps the most significant contribution of Taguchi is the fact that he simplified experimental design and popularised it. This can be seen by the large number of papers, books and webpages published about the method, its applications and modifications, e.g. (Kacker and Shoemaker, 1986), (Phadke, 1986), (Greenall, 1989), (Hamada, 1990), (Roy, 1990), (Vinning and Myers,1990), (Freeny and Nair, 1992), (Hamada, 1995).

2.1.4 Evolutionary Operation

Evolutionary operation (EVOP) is a simple factorial based experiment that was developed by Box (1957) in the late fifties, for application in the chemical industry. Although not strictly evolutionary in the currently understood sense, EVOP was an early version of a search technique based on natural processes (Goldberg, 1988b). EVOP was designed to be a simple and systematic method for continuous on-line improvement of a process, to be performed by the plant operatives. By methodically changing the operating parameters of a response, the surface can be plotted, giving a greater understanding of the process and hence the optimum operating conditions. The parameters should only be altered by small amounts so as not to greatly disturb the process, but this means that the experiment needs to be repeated several times before determining if a parameter has a significant effect. Based on simple factorial designs of experiments, EVOP is a fairly simple method of experimentation and the results are easy to display graphically.

For example, factors A and B are considered to influence the profitability of a process, so an EVOP experiment is drawn up to optimise these factors. The current operating conditions are as shown as 1 in Figure 2.2. The amount by which a factor should be changed can be difficult to determine as this change should be small enough so as not to greatly disturb the process yet large enough for any effect to be analysed.

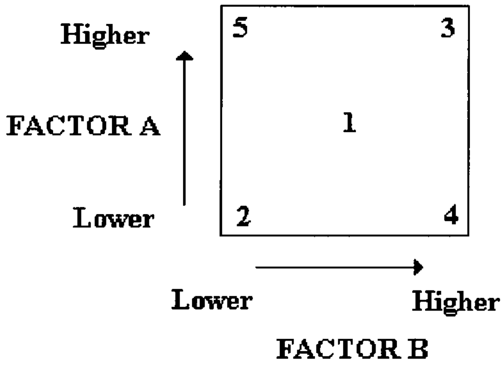


Figure 2.2. Order of runs of simple two factor EVOP.

The experiment is run in cycles, each cycle consists of five experiments, or runs, which are shown in Figure 2.2. After completion of the second cycle there is sufficient information to be statistically analysed to test if changing the level of a factor significantly affects the process. The experiment is run continuously until a factor is shown to influence the process. One phase is then said to be complete and the operating conditions are moved in the direction of the improved yield, often using the most promising point from the last phase as the central point of the new phase. A general rule of EVOP is that at least two points in the new phase cycle should have been part of the previous phase cycle, this prevents the process from rapidly moving into a much less profitable region. Box described this continuous movement as the evolutionary part of EVOP. There have been several modifications to EVOP, e.g. Rotating Square EVOP, Random EVOP (Lowe, 1964) and the most popular modified version Simplex EVOP (Lowe, 1964), (Lowe, 1974) but the original version seems to be the most widely used (Hunter and Kittrell, 1966).

EVOP is a simple method of process improvement, yet it has not been extensively used. One of the original reasons for managements' reluctance to implement EVOP (Hahn and Dershowitz, 1974) was its then revolutionary approach in promoting discussion groups including both operators and managers (Chatto and Kennard, 1961), (anonymous, 1961). This practise, is now widely accepted, largely due to the quality movement of the recent decade and the influence of Taguchi (1987). EVOP actively disturbs a process which could be seen as currently satisfactory (Lowe 1974), and although the ultimate aim is process improvement many plants were not prepared to risk producing sub-standard product whilst actively seeking greater understanding of the process (Hunter, 1989). Probably the main argument against EVOP when it was originally published was the amount of paperwork involved with the highly repetitive calculations. EVOP appears to have almost died out before the advent of relatively cheap computing power, which coupled with current shop floor data collection techniques could totally automate the process. Integrating with an expert system or rule-base could automate the system further. The main optimisation argument against EVOP is that it is a local search technique and as such it can get trapped in local minima. EVOP is not capable of global search.

Despite these shortcomings there are many examples of EVOP being used as originally intended in the chemical and process industries (Carleysmith, 1994), (Floudas and Anastasiadis, 1988), (Muraki et al, 1986), (Barnett, 1960) and a small amount of literature available on EVOP being used in other industries (Box and Draper, 1969), e.g. die casting (Chen, 1989). EVOP is mentioned in the new automotive industry standard QS-9000, indicating that its potential has been recognised by the three main automotive producers in the USA, Ford, Chrysler and General Motors, yet there seems to be little current use of this simple but effective technique. A more detailed explanation of EVOP can be found in Chapter 3.

2.1.5 Current Experimental Design

As discussed earlier in section 2.1.1. experimental design has experienced a renaissance in the last two decades. It has become more widely known and applied in a much wider variety of industries than the traditional chemical and process industries. There is a large volume of literature relating to experimental design and more general quality issues. Probably due to the emergence of global quality standards, such as ISO9000, there has been an increase in interest of quality issues in general, experimental designs and other techniques that can help companies achieve the required quality levels. Unfortunately the quality movement of the Eighties did not bring with it a panacea to industry's ills and the philosophy of the Nineties appears to be that of down-sizing and re-engineering. Many of the methods used are still valid useful tools for quality improvement and maintenance as product quality has become less of an issue and more of an expectation.

2.2 EVOLUTIONARY COMPUTATION

2.2.1 Introduction

Artificial intelligence (AI) is a large field which is currently experiencing much growth, after decades of often widely unnoticed progress. Evolutionary computation consists of many areas including genetic algorithms, evolutionary strategies, evolutionary programming, simulated annealing and genetic programming. All of these methods use evolution as a paradigm, but in different ways. There follows a brief overview of some of the methods included in the field of evolutionary computation.

2.2.2 Evolutionary Programming

Evolutionary programming (EP) was developed in the mid sixties in America by L. Fogel et al (1966), see Fogel, D.B., (1994) for a list of early papers. EP did not try to directly emulate the human brain, as much previous work had tried to do, but to model the process of evolution. EP initially assumes that the region is bounded but afterwards this restriction is lifted. Unlike other evolutionary computation methods such as genetic algorithms and evolutionary strategies, EP uses only mutation as an operator. This is the main 'biological' difference of EP. Each parent in a population is mutated once to produce an equal number of offspring. The entire population of parents and offspring is ranked according to fitness and then the fittest are selected to become the next population. This is subject to the constant population size constraint.

2.2.3 Evolution Strategies

Evolution strategies (ES) were developed in Germany by Schwefel (1965) and Rechenberg (1973). Mutation is the main search operator and children can be formed by either of two different recombination mechanisms; randomly select two parents to produce a new string; genes can be taken from the entire population to form a new individual. Selection is completely deterministic and extinctive, no probabilities of reproduction are used and only the best offspring are selected. The population size is restricted.

Originally ES had one parent for each child as in Renchenbergs' (1+1)-ES. Both parents and children competed for survival and the poorest solutions discarded. Problems with this method include slow convergence due to the constant step size and it has the brittleness of a search that moves from point to point, meaning that it can become trapped in local minima. Renchenberg proposed the first multi-membered ES, (μ+1)-ES, in which children were produced from more than one parent. Each time the child replaces the worst parent solution in the population, similar to the Simplex method mentioned previously. This form of ES was not widely used but it formed the basis for the later work of Schwefel (1975), who proposed using multiple parents (μ) and children (λ) in the ($\mu+\lambda$)-ES and the (μ,λ)-ES. In ($\mu+\lambda$)-ES all solutions compete for survival whereas in (μ,λ)-ES only the children compete.

ES is still an area of active research. Introductory information can be found in Back and Schwefel (1993), Fogel, D.B., (1994) and journals such as Evolutionary Computation (De Jong, 1993).

2.2.4 Simulated Annealing

Simulated annealing (SA) originates from the work of Kirkpatrick et al.(1983). The original work was in a 1982 IBM research report RC9355, according to van Laarhoven and Aarts (1987), but it was given a wider audience a year later in Kirkpatrick et al.(1983). SA is drawn from thermodynamics where annealing is the process of heating a solid to a high temperature, such that the molecules have a high energy. The solid is then cooled slowly until the molecules reach a low energy ground state to give a very pure crystal. Cooling is done slowly so that thermal equilibrium can be maintained. If cooling is done too quickly, the effect is known as tempering or quenching, the outside of the solid cools much faster than the centre which gives rise to large internal forces and faults in the lattice structure.

At each temperature the solid is allowed to reach thermal equilibrium, which can be characterised by the Boltzmann distribution. The probability of being in a state with energy E is given by:

$$\Pr\{\mathbf{E} = E\} = \frac{1}{Z(T)} \cdot \exp \frac{-E}{k_B T} \qquad \text{Equation 2.5}$$

where T = temperature,

k_B = Boltzmann constant

Z(T) = normalisation or partition factor

The solid is then cooled again and the process repeated until the molecules form a lattice as the minimum energy state is reached.

"Simulated annealing is a stochastic computational technique derived from statistical mechanics for finding near globally-minimum-cost solutions to large optimization problems." (Davis, 1987). The parallels with physical annealing are as follows, the energy function becomes the objective or cost function, C, temperature becomes the control parameter, c, the lower the energy the better the solution. The parallel with thermal equilibrium is achieved by the Monte Carlo method which is also known as the Metropolis algorithm. A small disturbance is given to a random particle, configuration, and the change in energy states compared. If the new state has a lower energy then it is accepted as the new state. The system evolves to a state of thermal equilibrium and the probability of states approaches the Boltzmann distribution. In SA terms this means a sequence of Metropolis algorithms are evaluated at a series of decreasing values of the control parameter. The process continues until equilibrium is reached and the probability distribution of the configurations approaches the Boltzmann distribution, given by:

$$\Pr\{\text{config.}=1\} = qi(c) = \frac{1}{Q(c)} \cdot \exp\frac{-C(i)}{c} \qquad \text{Equation 2.6}$$

'Cooling' is carried out and the process repeated until 'freezing' occurs, where the change in the cost parameter, c, is virtually non existent and near the optimum solution. As with annealing SA needs a schedule to determine at which temperatures the solid should be held and the duration.

The need to maintain thermal equilibrium means that annealing is inherently slow, but parallel processing can speed up simulated annealing to acceptable times. SAs have been applied to many areas including computer design, image segregation and restoration, the travelling salesman problem and artificial intelligence, for example see (Bonomi and Lutton, 1984), (Kirkpatrick et al. 1983), (Davis, 1987) and (Van Laarhoven. and Aarts, 1987).

2.2.5 Genetic Programming

Genetic programming (GP) is based on the principles of natural selection or Darwinism and genetic operators, which is similar to the basis of GAs, but the paradigm is applied specifically to the creation of computer programs. GP is sometimes referred to as 'automatic programming' (anonymous, 1999) as the populations of GP consists of sets of programs that are candidate solutions and the final 'solution' is a program rather than a encoded string. This form negates the need for much pre and post processing of inputs and outputs of a GP. A typical individual is shown in figure 2.3.

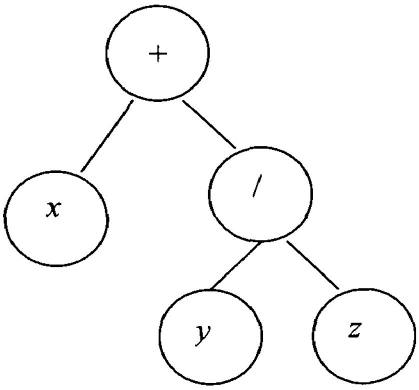


Figure 2.3. Typical individual in a GP population (representing x + y/z)

The programs are composed from elements in a function set and a terminal set. The function set typically comprises of the operators that generate the model, i.e. the functions label the internal points of the parse trees representing a program in the population, e.g. in figure 2.3 the function set would include '+' and '/'. The terminal set comprises of the terminal, or leaf, nodes in the parse trees that represent the programs in the population, e.g. in figure 2.3 the terminal set would include 'x', 'y' and 'z'. The terminal set may compromise of variables, as shown above, but may also include constants and functions that have no arguments.

The fitness of individuals in the population is evaluated in terms of performance in the required problem environment and the fittest individuals will have a higher probability of surviving and reproducing.

There is often a form of elitism where the fittest individual programs are retain by replicating the individuals into the new generation. The other method of producing a new generation is by crossover. Crossover is implemented by taking a randomly selected sub-tree of a fitness selected individual and exchanging that sub-tree with another in another individual. The newly formed generation is then evaluated for fitness before reproduction is repeated, until a predetermined end point, such as level of fitness or number of iterations.

Traditionally GP does not usually use the mutation operator (Koza, 1992), unlike GAs and in sharp contrast with EP. When mutation is applied within GP it is generally with a low probability rate and it can be used at various levels, e.g. a sub-tree of a program is deleted and replaced by a new randomly grown sub-tree (Fernandez, 2000) or mutation could be carried out on only one node of a tree solution. (Alvarez, 2000).

2.2.6 Genetic Algorithms

Genetic algorithms (GAs) stem from the work of John Holland (1992) originally published in 1975. Like EVOP, GAs are based on the principles of natural evolution, but more closely follow what the currently understood method of natural reproduction but at a genetic rather than species level. Unlike EVOP, GAs are a robust, global search method. GAs vary from most traditional search methods in that they search a population of points in parallel rather than exploring from a single point, they do not need any *apriori* knowledge of the problem, use probabilistic rather than deterministic rules and usually work on a set of encoded rather than real world variable values.

Information about the environment is encoded as a string, traditionally in binary form. The search usually starts from a series of random points and the fitness of the solutions at these points is evaluated using an objective function. The next generation is produced using three main operators; selection, crossover and mutation. There are many methods of selecting which parent strings should enter the mating pool: Techniques include roulette wheel selection where the fittest solutions are given the highest probability of reproduction; elitism strategies where the fittest solutions are guaranteed passage into the next generation; methods based on ranking the comparative fitness of the solutions on sliding scales.

Once selection is completed formation of the next generation can take place. Reproduction is implemented using crossover. One point crossover is the simplest form; two parent strings are 'cut' at the same point along their length and the 'tails' swapped, making two new strings. Variations include two point crossover and uniform crossover where templates are used (Syswerda, 1989).

One important concept with crossover is schemata. These are short parts of the string which are highly fit, they are also known as building blocks as it is the aim to retain these highly fit blocks within strings in the next generation.

Mutation is the final main operator. It usually occurs with a very low probability, one in a thousand say. Mutation changes the value of a bit in the string, e.g. 0 to 1 or vice versa for binary strings, and is known as the GA 'insurance policy' as it is always possible to reach any part of the search space whatever the starting points. There are many alternatives to determine if newly generated 'child' strings should join or replace the current population. The new population is then evaluated. If the number of generations allowed is not exceeded and the solution generated is not suitable, the new population is then used to create the next generation.

The interest in GAs is reflected in the increasing amount of work published, for surveys see (De Jong, 1993, Srinivas and Patnaik, 1994, Caponnetto et al, 1993), the ever increasing number of evolutionary computation conferences and in the publication, since 1993, of the journal titled Evolutionary Computation. There are also many sites on the Internet devoted to evolutionary computation, for example The Hitch Hiker's Guide to Evolutionary Computation (Heittkötter and Beasley, 2000).

GAs are excellent global search methods but convergence can be a problem as can the inability to converge on a solution. Having located a highly fit region, GAs can have problems locating a local optimum. It is the inability to search effectively at a local level that is a major drawback of GAs. Work has been carried out in this area (Kwong et al, 1995), and research into hybridising GAs with other search techniques (Kido et al, 1993), especially local search methods, for example (Renders and Bersini, 1994). The work in this thesis will address this issue, but in contrast with the papers cited above, by using small population GAs.

Another problem with GAs is the amount of time it can take to arrive at an acceptable solution. This is due to the size of the population, which is often quite large, and the number of generations needed. Reducing the population speeds up the creation of a new generation but not as much of the search space is covered. There appears to be very little work published in this area (Krishnakumar 1989, Reeves, 1993), and this thesis investigates this issue.

2.3 SUMMARY OF THE SELECTION OF METHODS FOR EXPERIMENTATION

Experimental design is well established but often only used to its full potential in its originally intended fields such as industrial chemistry. There is a movement to encourage exploitation of these methods in the industrial scenario

(Hamada, 1995).

The simplex method gives good final results but moves around a search space quickly and is prone to give sub-standard answers during the search, which could potentially be costly for industrial applications. The Taguchi method has some flaws, but the concepts and philosophy are sound, as discussed in section 2.1.3.7. These methods were felt to be too complex to allow significant savings in genetic algorithm run-time, but Taguchi has proved the positive aspects of experimental design for industrial use.

EVOP is simple effective method which suffers from a lack of use and current research, but does offer a potential for improving a genetic algorithm search without costly increases in run-time or processing power.

Evolutionary Programming and Evolutionary Strategies are both forms of genetic computation, but have not been researched to the same extent as genetic algorithms and industrial applications are not as abundant. Simulated Annealing is based on thermodynamics, and the literature indicated it as an effective but inherently slow method, therefore unsuitable for on-line applications. Genetic programming is used for very specific applications in the generation of software code.

Genetic Algorithms are becoming an established field, with much active research. Although there are many papers published on GAs relatively few are published in the area of hybrid GAs. To produce a fast, robust optimisation method, with both global and local search capabilities, GAs and EVOP were chosen for further study to produce a hybrid GAs that is reliable and yet quick, to be applicable and attractive to industry to use on-line. These techniques, their positive and negative aspects and the software available to implement the searches are discussed in further detail in the next chapter, chapter three.

3. EVOLUTIONARY OPERATION & GENETIC ALGORITHMS

The aim of this chapter is to give a deeper understanding of evolutionary operation and genetic algorithms, the two methods selected as discussed in chapter two, which are to produce new hybrid methods. Details of applications of these methods are given and the availability of software discussed. A review of genetic algorithm hybrids is then undertaken before describing the selection of software for the hybrid methods tested in this thesis.

3.1 EVOLUTIONARY OPERATION

3.1.1 Example

As discussed previously in chapter two Evolutionary Operation (EVOP) is a simple factorial experimentation technique developed by George Box in the late 1950's. Initially intended for use in the chemical process industry, EVOP has since been applied to other industries, such as die casting (Chen, 1989). EVOP is designed for continuous on-line improvement to be implemented by the plant operatives. The method continuously searches the local response surface area by altering the process parameters by small amounts, hence not greatly disturbing the process or producing sub-standard products.

The EVOP method will now be described through an example. Information in Barnett (1960) has been modified to provide the example which is based on the study of the yield of a chemical process. The important factors are thought to be temperature and pressure. The experiment is run in cycles, in this case each cycle consists of five runs, as shown in Figure 3.1.

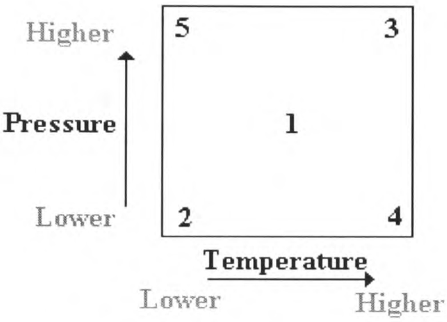


Figure 3.1. Five runs of an EVOP cycle.

In an EVOP experiment the cycle is repeated until there is a significant improvement in yield due to one or more of the factors being studied, at this stage one phase is said to be complete. The next phase of cycles is then started, usually with the improved yield setting being used as the centre point for the new design. If after a pre-determined number of runs there is no significant improvement in the yield then the experiment should be halted as studying other factors may be more productive.

The amount of information generated by the first cycle is insufficient to calculate the standard deviation and to indicate any significant factors. For this reason the first cycle (N=1) of the example is considered complete and the data from that cycle inserted as previous cycle information in the table for the second cycle (N=2) as shown in Table 3.1.

Cycle 2 (N = 2)

Operating Conditions	**1**	**2**	**3**	**4**	**5**
Sum from previous cycle	9.38	6.66	11.04	9.04	9.48
Av. From previous cycle	9.38	6.66	11.04	9.04	9.48
New Observations	7.92	8.40	8.83	8.25	9.96
Differences	+1.46	-1.74	+2.21	+0.79	-0.48
New sum	17.30	15.06	19.87	17.29	19.44
New average	8.65	7.53	9.94	8.65	9.72

Range of new observations = |+2.21| + |-1.74| = 3.95

Table 3.1. Observations and initial calculations for cycle 2.

Operating conditions correspond to those defined in Figure 3.1. The fourth row of Table 3.1 shows the readings for the current cycle. The fifth row is the difference between the observed reading and the average readings of previous cycles; care should be taken with the signs, as this basic mistake has been found in published papers. The last two rows show the new sum of observations and the new average, these then form the second and third rows of the table for the next cycle. The constants used for the calculation of the standard deviation and error limits are shown in Table 3.2. These constants are used to simplify the calculation for standard deviation based on the range of the observations. This provides an unbiased estimate of σ. With a sample size of more than ten 'the efficiency of the range method falls off' (Box and Draper, 1969), but by randomly dividing the sample up into subsamples of ten or less and using the average of ranges, an estimate of σ can be found. The usual estimate has a slight bias. Derivation of the constants can be found in Box and Draper (1969) or derived using standard statistical tables.

N	K	L	M
2	0.30	1.96	1.76
3	0.35	1.33	1.19
4	0.37	1.09	0.96
5	0.38	0.95	0.85
6	0.39	0.85	0.76
7	0.40	0.78	0.70
8	0.40	0.72	0.65

Table 3.2. Constants used in EVOP calculations

Calculation of standard deviation:

Previous sum = *0*

New sum (for cycle 2) = *Range of new observations* x *K*
= *3.95* x *0.30*
= *1.185*

New sum all cycles = *Previous sum + New sum (for cycle 2)*
= *0 + 1.185*
= *1.185*

New average = *New sum / (N-1)*
= *1.185 / (2-1)*
= *1.185*

Previous average (for cycle 1) = *0*

Calculation of 95% error limits:

For new averages and effects = L x *New sum*
=*1.96* x *1.185*
= ±*2.32*

For change in mean effect = M x *New sum*
= *1.76* x *1.185*
= ±*2.09*

To determine the effects of factors the observations of a cycle are entered into rows 1 and 2 of an EVOP table following the scheme of Table 3.3: The letters correspond to Figure 3.2. To calculate the effects of factors A, B, interaction AB and a positive change in mean effect the first two rows are added together to produce row 3. For a negative change in mean effect the centre point reading is multiplied by four and entered in that column at row 3. For each effect the largest and smallest totals in row 3 are determined and each smaller total is placed in row 4 directly below the larger total. The difference in these totals is then calculated by subtracting the value in row 4 from the value in row 3, the answer being placed directly below in row 5. For factors and interaction effects the values in row 5 are divided by two and the answer placed directly below in row 6. For changes in mean effect the value in row 5 is divided by four and the answer placed directly below in row 6. Whether the effect is positive or negative is determined by the column of the sixth row value.

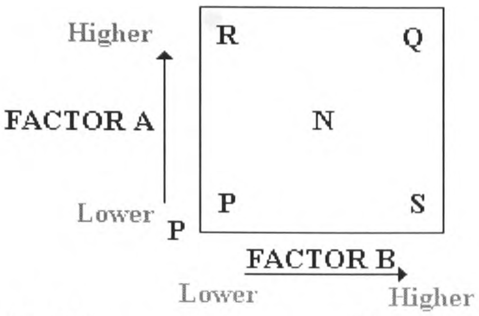


Figure 3.2. Scheme of letters used in Table 3.3.

Row Number	**Effect of A**		**Effect of B**		**Effect of AB**		**Effect of CIM**	
	+	-	+	-	+	-	+	-
1	R	P	Q	R	Q	R	P + S	N
2	Q	S	S	P	P	S	R + Q	x 4
3	R + Q	P + S	Q + S	R + P	Q + P	R + S	P +S+ R+ Q	4(N)
4								
5								
6								

Table 3.3. Calculations for assessing effects.

Returning to the example experiment, the effect of pressure, see Table 3.4, can be calculated by comparing the average readings taken at the lower pressure, positions 2 and 4, with those at the higher pressure, positions 3 and 5 of the cycle, see Figure 3.1 and Table 3.1. Comparing the differences in the sums at the lower and higher positions indicates the magnitude of the effect. For each factor in Table 3.4 the positive effect is the left hand column. For the change in mean effect the sum of the readings of the four outer points (2,3,4,5) are compared to the central point reading multiplied by four. It can be seen in table 3.4 that none of the values of the effects are larger than the error limits, ±2.32 for temperature and pressure effects or ±2.09 for the change in mean. Therefore another EVOP cycle should be run. The results of the next cycle are shown in table 3.5.

Calculation of effects:

	Effect of A		Effect of B		Effect of AB		Effect of CIM	
Row	+	-	+	-	+	-	+	-
1	9.94	7.53	9.94	7.53	7.53	8.65	18.59	8.65
2	8.65	9.72	9.72	8.65	9.94	9.72	17.25	x 4
3	18.59	17.25	19.66	16.18	17.47	18.37	35.84	34.60
4	17.25		16.18			17.47	34.60	
5	1.34		3.48			0.90	1.24	
6	+0.67		+1.74			-0.45	+0.25	

KEY **A** = pressure **AB** = interaction of temperature and pressure
B = temperature **CIM** = total change in mean

Table 3.4. Calculation of effects for cycle 2.

Cycle 3 (N = 3)

Operating Conditions	**1**	**2**	**3**	**4**	**5**
Sum from previous cycles	17.30	15.06	19.87	17.29	19.44
Av. From previous cycles	8.65	7.53	9.94	8.65	9.72
New Observations	9.34	8.53	10.66	8.52	9.53
Differences	-0.69	-1.00	-0.72	+0.13	+0.19
New sum	26.64	23.59	30.53	25.81	28.97
New average	8.88	7.86	10.18	8.60	9.66

Range of new observations = |+0.19| + |-1.00| = 1.19

Table 3.5. Observations and initial calculations for cycle 3.

There is a need to calculate a new standard deviation and error limits based on the new observations

Calculation of standard deviation:

Previous sum = *1.185*

New sum (for cycle 2) = *Range of new observations* x *K*
= *1.19* x *0.35*
= *0.417*

New sum all cycles = *Previous sum* + *New sum (for cycle 3)*
= *1.185* + *0.417*
= *1.602*

New average = *New sum / (N-1)*
= *1.602 / (3-1)*
= *0.801*

Previous average (for cycle 2) = *1.185*

Calculation of 95% error limits:

For new averages and effects = *L* x *New sum*
=*1.33* x *0.801*
= ±*1.065*

For change in mean effect = *M* x *New sum*
= *1.19* x *0.801*
= ±*0.953*

To be significant the effects of temperature and pressure need to be greater than ±1.065 and the change in mean effect greater than ±0.953. It can be seen in Table 3.6 the effect B, temperature, is significant as +1.69 ±1.065 does not equal or include zero in its range. This indicates that a higher temperature would produce a better yield. One phase is now complete.

Calculation of effects:

	Effect of A		**Effect of B**		**Effect of AB**		**Effect of CIM**	
Row	+	-	+	-	+	-	+	-
1	10.18	7.86	10.18	7.86	7.86	8.60	18.78	8.88
2	8.60	9.66	9.66	8.60	10.18	9.66	17.52	x 4
3	18.78	17.52	19.84	16.46	18.04	18.26	36.30	35.52
4	17.52		16.46			18.04	35.52	
5	1.26		3.38			0.22	0.78	
6	+0.63		+1.69			-0.11	+0.15	

KEY **A** = pressure **AB** = interaction of temperature and pressure
B = temperature **CIM** = total change in mean

Table 3.6. Calculation of effects in cycle 3.

Traditionally at this point an EVOP committee, consisting of plant operators, process specialists and statisticians, would meet to discuss the findings and decide upon the factors and settings to be studied in the next phase. An application algorithm could be devised to speed up this step, but the EVOP team should still met periodically to review progress. As with other factorials the order of runs should be random, but sometimes the nature of the process forces a particular order, this should be taken into account when analysing the data. If all the points around the centre indicate a drop in yield then this part of the experiment should be regarded as accomplished, as the local maximum has been reach, and new ideas should be explored. EVOP is designed to be continuously in operation, so should never be regarded as complete.

3.1.2 EVOP Philosophy

EVOP, like the more modern Taguchi method, also encompasses the ethos of teamwork as well as the experimental and statistical elements. Awareness of EVOP should be throughout the company from shop floor operators to the highest level of management. This viewpoint that all levels of employees should regularly participate in process improvement was revolutionary when first introduced; contemporary managers are usually more familiar with multilevel communication and an inter-disciplinary team approach.

3.1.3 EVOP Modifications

There are three notable modifications to EVOP, although none appear to have been as widely used as the original design. Lowe (1964) discusses all three, rotating square EVOP (ROVOP), random EVOP (REVOP) and simplex EVOP.

"ROVOP attempts to eliminate uncertainty because of the size of the variant used." (Lowe 1964). The initial experimental design is a 2^N factorial with a centre point. For each complete cycle the variant, or factor, step size is enlarged by √2 and the design rotated by 45°. This is shown in Figure 3.3 below, the internal square represents the first cycle and the outer, the second rotated and enlarged cycle.

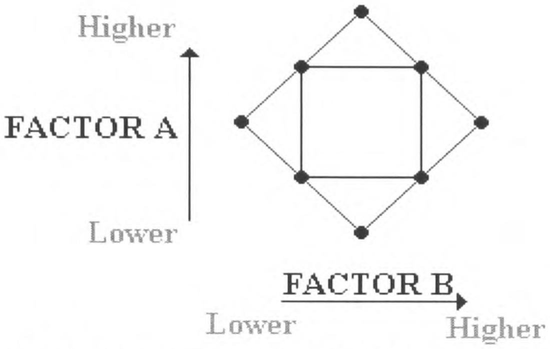


Figure 3.3. Rotating square EVOP (ROVOP).

The design is rotated and enlarged in this manner until a direction of improved yield is indicated. The experiment is then moved in this direction. At least two points of the new design should have previously been explored, and the size of this new cycle is at least a factor of √2 smaller than the previous design. Once straddling an optimum, the design can be reduced in size to minimise factor variation, whilst allowing process information to be generated. This method claims to cover a response surface more rapidly than EVOP and the starting levels of the factors are not important, but a more complex analysis using multiple regression is required where all factors must be quantitative. For more than three factors the design becomes complex.

Another proposed variation of EVOP is random EVOP (REVOP). It is designed for experimentation with a large numbers of factors with unknown relationships and forecasting reactions. The choice of data points is random and the step size is kept at 20% of the range of the factor under investigation. If there is an improvement then the next step is taken in the same direction. If there is no improvement then a new set of random points is chosen. The advantages of this method is that any number of factors can easily be incorporated and the calculations are very simple. There are many disadvantages including the random movement of the design, factors must be quantitative, the step lengths may not be practical, there is no separation of effects from the factors and no response surface can be built from this method.

The final, most popular modification is Simplex EVOP, first presented by Spendley *et al* (1962). This method was developed from a desire to speed up EVOP and to execute it on a computer: it should be noted that this paper was published in 1962 when computers were not as powerful or as widely available. Rather than a sequence of full factorials a succession of regular simplex designs are used. A more detailed description of the simplex method can be found in section 2.1.4. of this thesis. Simplex EVOP can be summarised as shown in figure 3.4.

Once simplex EVOP has found an optimum it will circle around a fixed optimum or follow a continuously moving optimum. Benefits of this method include the use of only the most recent and therefore most relevant data, continuous movement so false moves are quickly corrected and the method does not require complex mathematics, merely comparison of data to reject the worst

```
Make three initial observations
Eliminate the worst observation
Make another single observation in the most favourable direction
    move after every subsequent observation by rejecting the worst point,
    unless
    another observation is too old, then renew this observation,
    or
    the move means returning to a previous point, then move to next most
    favourable direction.
```

Figure 3.4. Summary of Simplex EVOP

point. The method has drawbacks in that the factors must be quantitative, the factor variants must be of equal interest to the experimenter, which is not always the case, to ensure a regular simplex and finally a suitable partition can be very difficult. Factors can easily be added, but deleting a factor in a running simplex EVOP is not possible, a new experiment must be started.

A move is made after every observation and "Thus the simplex approach is more dynamic and also less conservative (and often less informative) than the classical factorial approach." (Hahn and Dershowitz, 1974). Simplex EVOP can be rotated "...and in the presence of error and without replication it is the most efficient design." (Lowe, 1964).

3.1.4 Demise of EVOP and Reasons for its Lack of Use

Although EVOP is relatively simple in implementation and avoids complex statistics, it requires a large number of repetitive calculations. In the 1950's and early 1960's, when EVOP was introduced, the calculations were probably seen as a laborious task by any process supervisor designated to perform them. The advent of powerful, relatively cheap computing power now allows the calculations to be performed automatically. Automated shop floor data collection can reduce the task even further so that all that is required is periodic discussion of the results. But as J.S. Hunter (1989) pointed out " EVOP has not become an active weapon in the armoury of many of today's quality experts", this he attributed to the complexity of EVOP procedures and the need for "an active mind set", not passive observation of the process. The main advantage of EVOP is that it advocates minimum disruption of the output. It could be applied to more factors, although traditionally it was restricted to three as it relied heavily on visualisation and studying more factors can obscure interaction effects. Again computer software can help to overcome this problem. Although not conventionally directly linked to process improvement, employee and managerial participation can mean that communication is improved and employees feel that their opinions are valued. Conversely if the experiment does not produce an improvement in the process it could affect the morale of participants who may then see EVOP as a worthless exercise, and hence place barriers to further EVOP experimentation.

3.1.5 Reviews of EVOP

A good introduction to EVOP, indications of suitable applications and its place on the shop floor are given in Chatto and Kennard (1961). EVOP is recommended for investigating processes where there is a lack of theoretical knowledge and "Therefore the real advantage of the technique is that it promotes a new way of thinking, and utilizes to its fullest extent the human factor." (anonymous, 1961). The argument behind this statement is that processes and factors may change, but workers are still required and they are more effective and capable if they are accustomed and receptive to change. EVOP training programmes also reinforce this as employees see an investment in themselves as well as the process. Hunter and Kitrell (1966) in their review of EVOP publications identified three major overlapping areas; methodology, applications and modifications. There are many references to applications of EVOP, e.g. at ICI and American Cyanamid Company, but it reinforces the opinion that although EVOP has been used in the chemical industry, there is very little evidence of its application to other areas.

Hahn and Dershowitz (1974) describe high volume processes with relatively long run times, easily varied factors, which are quick to stabilise after disturbance and where the potential benefits are thought to be large, as the most suitable for EVOP. Also needed is knowledge of the important factors and a clear definition of the response to be maximised. A brief survey was also presented of the nature and extent of EVOP in industry. It showed that EVOP was not as widely used as it could be. Reasons given included that it was not seen as profitable to vary process factors, a general ‘reluctance to perturb the manufacturing process’ and ‘political reasons’. Hahn and Dershowitz (1974) concluded, as Hunter (1989), that EVOP could be much more widely applied. Lowe (1974) comments on possible reasons for the lack of EVOP applied in industry. It was found that there was a reluctance by supervisors to accept disturbances to their processes when there is any chance of sub-standard product resulting. Also supervisors viewed their experience more valuable than statistical testing. Lowe also gave examples of, and discussed EVOP and simplex EVOP, and to explain their merits: “What should be realized is that, for every process working at less than theoretical yield, some improvement is possible and evolutionary operation studies are the most painless ways of exploring the possibilities.”

Box, the originator of EVOP, has expressed opinions about the modifications to his original method (Box and Draper, 1969). ROVOP is designed to be used when the EVOP team is too cautious and conservative in its choice of factor levels, but Box maintains that it is best to use standard EVOP and leave the team to freely choose their next move. EVOP is probably slower than ROVOP which could easily be implemented as a computer program because it has well defined rules. REVOP is dismissed as “...Our own experience suggests that there is very little to recommend this procedure...” as it is very difficult to change more than three factors in a phase and visualisation is impossible, which breaks the EVOP philosophy of simplicity. Simplex EVOP is recommended for numerical optimisation but not process optimisation. The lack of repetition means that significant effects cannot be found to enable a greater understanding of the process. Box indicates the mechanical nature of the decision making, discarding the worst point to determine the next experimental area, without "scientific feedback", i.e. often valuable comments from the EVOP team, also disputed is the claim that simplex EVOP is quicker than standard EVOP.

3.1.6 Current Use of EVOP

EVOP appears to be an under used technique. There is evidence of its use in the chemical industry, but little elsewhere, see section 3.1.1. EVOP was a method before its time in many respects. The repetitive nature of the method seems to have contributed to its demise alongside the management–worker team approach, which would now be seen as a positive aspect of the method and repetitive calculations are the ideal work of computers.

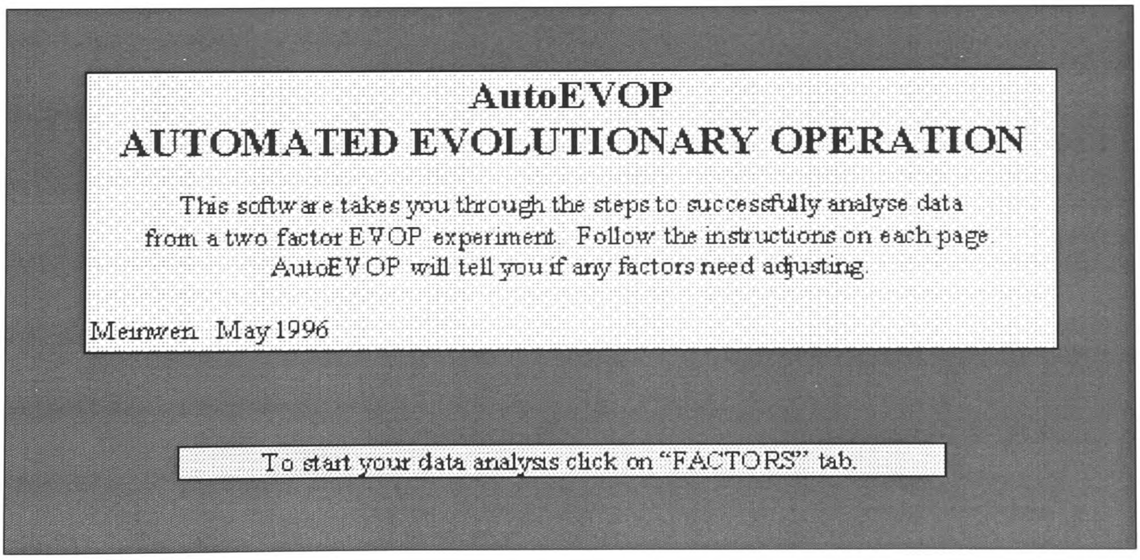
Until very recently there was no commercially available software to be found. This may change with the introduction of QS-9000, the quality standard for the automotive industry, created by the 'big three' producers in the USA, Ford, Chrysler and General Motors, for all their suppliers of production and service parts and materials world-wide. For design suppliers, both classical and Taguchi experimental design techniques are required skills. For all suppliers, continuous improvement is an expected company culture as is knowledge of various techniques to accomplish this, including EVOP, which are listed in the current edition of Quality System Requirements QS-9000.

EVOP is taught as part of some undergraduate courses, such as Industrial Statistics courses, but this is far from the ideal envisaged by Box that EVOP should be taught to every undergraduate engineer. There are short courses run on EVOP, such as 'EVOP: designed experiments for operating processes' by J.S. Hunter based at Princeton University. The emergence of EVOP as a topic could be in part due to QS-9000 and the greater prominence that experimental design now plays in industry. EVOP is also very briefly mentioned in recent publications such as Breyfogle (1999) in relation to six sigma and Park (1996) in relation to robust design.

3.1.7 AutoEVOP

At the time of initial implementation, no EVOP software could be located and all published work with worked examples of EVOP that was located, such as Barnett (1960), Box and Draper (1969) and Chen (1989) used hand filled worksheets. In response to the lack of software and to give a greater understanding of the technique, a new piece of software, AutoEVOP, was created as part of this study.

Initially a program was developed to run a two factor EVOP experiment in the C++ language. This gave an awareness of the difficulties of translating statistics into code. It became clear that EVOP would lend itself very easily to a spreadsheet application. Two and three factor EVOP programs have been written on Excel spreadsheets. Excel was chosen as a base application as it is widely available and used. All that is required to carry out the EVOP experiment is an understanding of the method and basic computer literacy. The initial screen seen by the user is shown in Figure 3.5. The second screen is used once the factors have been chosen. They are entered in the appropriate boxes, a table gives a description of the settings, see Figure 3.6. The experiment is run and the readings entered in the appropriate spaces, either manually or by automatic interface. Once two runs are complete and the data entered, AutoEVOP will give advice. Either to continue with the experiment or if a factor has been determined as significant recommend the best course of action, e.g. stop the current experiment and start a new one, see Figure 3.7. Locked cells in the spreadsheet prevent accidental damaged to the program by the user, allow data to be entered only in certain cells and prevent malicious hacking of the code. With appropriate automatic data collection, the EVOP experiment could become totally automatic.



AutoEVOP
AUTOMATED EVOLUTIONARY OPERATION

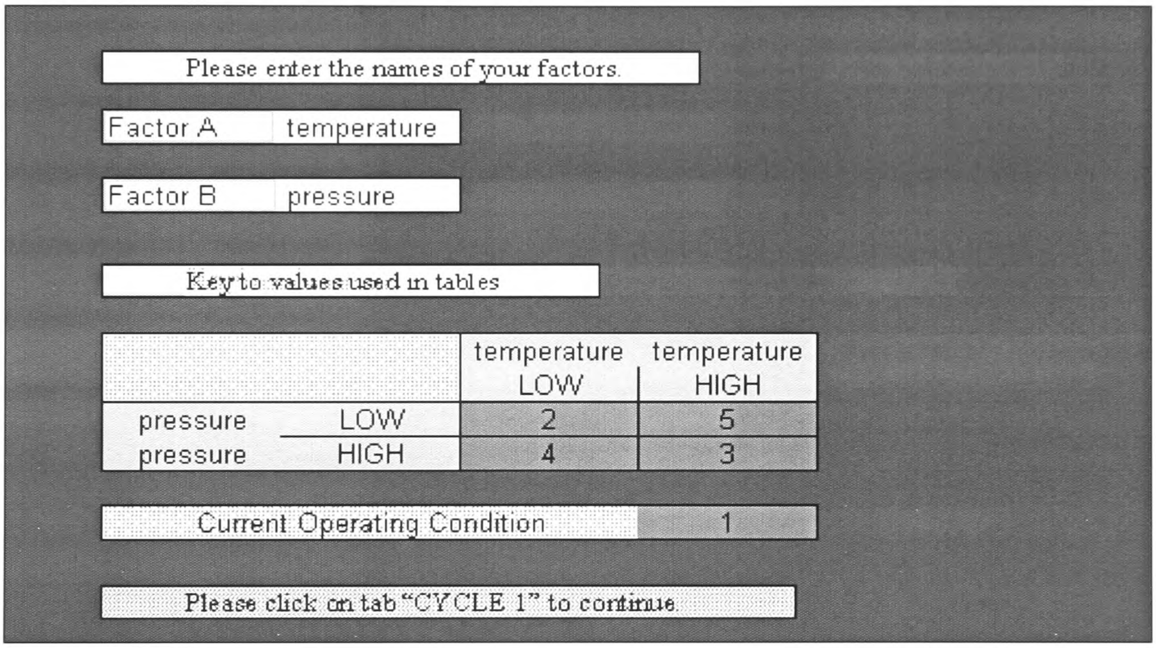
This software takes you through the steps to successfully analyse data from a two factor EVOP experiment. Follow the instructions on each page. AutoEVOP will tell you if any factors need adjusting.

Meinwen May 1996

To start your data analysis click on “FACTORS” tab.

Figure 3.5. Welcome screen from AutoEVOP.

For ease of use the interface is designed so that the user may only enter data into cells that are highlighted in blue and all user entered text appears as blue. For the screen shown in Figure 3.6, the user has entered 'temperature' and 'pressure' as the factors to be studied. These are the only cells available for the user to enter information.



Please enter the names of your factors.

Factor A	temperature
Factor B	pressure

Key to values used in tables

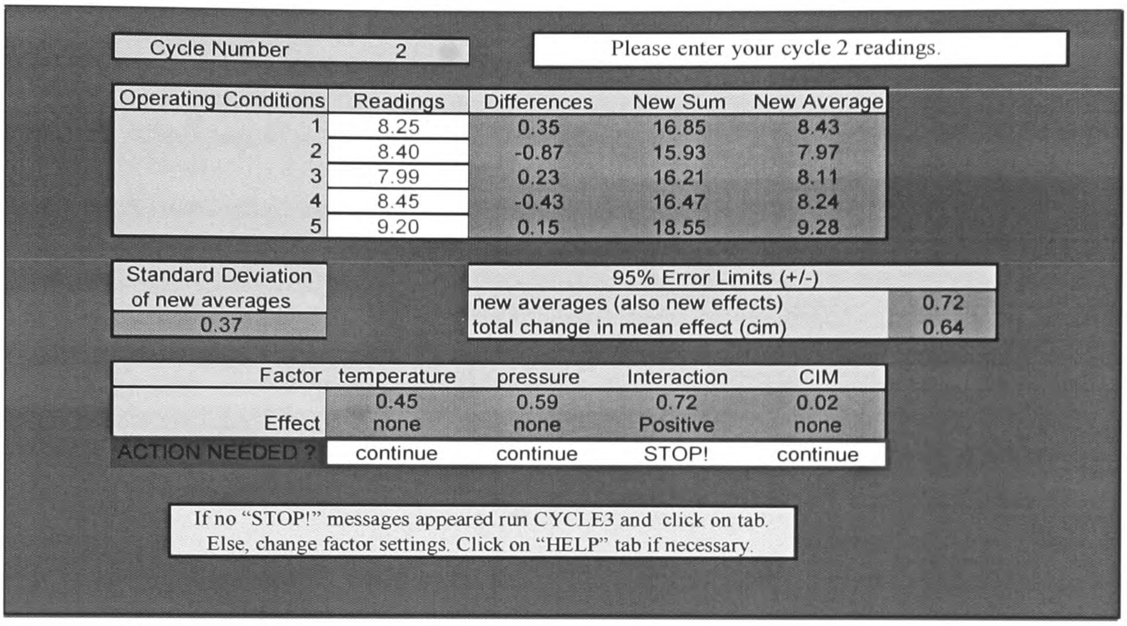
		temperature LOW	temperature HIGH
pressure	LOW	2	5
pressure	HIGH	4	3

Current Operating Condition 1

Please click on tab "CYCLE 1" to continue.

Figure 3.6. Entering of factor names.

In Figure 3.7 the user, or data entry device, may only insert data in the readings cells, also highlighted with blue text. The readings entered have produced a "Stop" situation and a positive interaction effect has been indicated. All cells that require user data to be entered are in blue and all action statements are highlighted in red which gives a simple interface for users. Interested parties can study the black text in the grey areas for further information if required.



Cycle Number	2

Please enter your cycle 2 readings.

Operating Conditions	Readings	Differences	New Sum	New Average
1	8.25	0.35	16.85	8.43
2	8.40	-0.87	15.93	7.97
3	7.99	0.23	16.21	8.11
4	8.45	-0.43	16.47	8.24
5	9.20	0.15	18.55	9.28

Standard Deviation of new averages
0.37

95% Error Limits (+/-)	
new averages (also new effects)	0.72
total change in mean effect (cim)	0.64

Factor	temperature	pressure	Interaction	CIM
Effect	0.45 none	0.59 none	0.72 Positive	0.02 none
ACTION NEEDED ?	continue	continue	STOP!	continue

If no "STOP!" messages appeared run CYCLE3 and click on tab.
Else, change factor settings. Click on "HELP" tab if necessary.

Figure 3.7. Typical cycle sheet.

3.2 GENETIC ALGORITHMS

Genetic algorithms (GAs) as stated earlier in chapter two, are based on natural genetics. The biological inspiration of GAs is reflected in the language used. A brief explanation of some of the terms (Goldberg, 1988, Fogel, D.B., 1994) is given below in Figure 3.8.

The GA is different from most traditional search methods in that the search is carried out from a population of points rather than from a single point. No a priori knowledge of the problem is required, probabilistic rules are used as opposed to deterministic, and the manipulation is usually carried out on encoded rather than real world variable values. GAs are a robust method of improvement. They work by blindly searching a population of points on a surface, but in a more efficient manner than pure random search methods or hill climbing methods which proceed step by step to a better adjacent space. Information is encoded in a string, traditionally in binary form. Strings are usually made up of many shorter sections each containing information about a factor.

Natural	***Genetic Algorithm***
chromosome	*string*
gene	*feature, character*
allele	*one of two alternative forms of a gene*
locus	*string position*
genotype	*structure*
phenotype	*expressed behavioural traits*
epistasis	*non linearity*
schemata	*string over an extended alphabet*
pleitropy	*a single gene may affect several phenotypic traits*
polygeny	*a single characteristic may be determined by many genes*

Figure 3.8. The language of genetic algorithms.

3.2.1 Schemata

Schemata are subsets of the design space which have attributes in common, e.g. in three dimensional space (x, y, z) a schemata could be: (1, *, *), where * represents a wild card. All points lying on the line 1 on the x axis would be part of the schemata set, e.g. the points (1, 2 ,3) and (1, 2, 1) would be part of the subset but the point (2, 1, 1) would not, see Figure 3.9.

The complete set of tuples belonging to (1, *, *) is called the set of schemata. "Schemata provide a basis for associating combinations of attributes with potential for improving current performance." (Holland, 1992). Schemata can be regarded as building blocks for GA strings. The best solutions contain the best schema, so early identification of good schema could allow best solutions to be found more quickly.

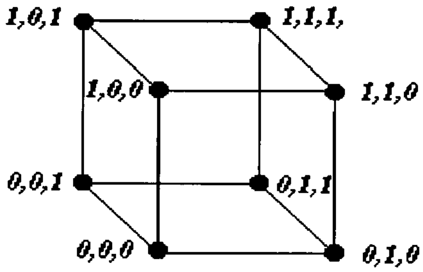


Figure 3.9. Three dimensional schema.

3.2.2 Generic Genetic Algorithm

A generic GA is listed in Figure 3.10, which outlines the basic steps of a GA. These steps will then be explained in more detail in the following sections.

Problem identified

Objective function (description of problem space)

**Candidate solution population*

Decode (to give fitness value for each solution) then re-code

Generate new population (by applying GA operators)

Decode (to give fitness value to each solution) then re-code

*If suitable solution or predefined number of generations produced stop, else go to **

Figure 3.10. Generic Genetic Algorithm.

3.2.2.1 Initial population

GAs are parallel search methods, so at any given time the GA has a group of possible solutions to a problem. This group of solutions is known as a population. GA populations tend to change as the GA progresses, as natural populations change over time.

An initial GA population of points is usually chosen randomly. Other methods for initial population selection include using known best solutions from historical data, but this requires that the problem has been solved previously. There is a small amount of research into other methods of selecting initial populations such as selecting individual solutions in the search space using an algorithm based on the Taguchi method (Lee and Rowlands, 1998).

A GA can take an excessive amount of time to find a feasible solution if the initial population is randomly selected, but if historic best known solution data is used then there is a risk of forcing the search to converge prematurely.

3.2.2.2 New population generation

The initial population is measured by a fitness function. The fitness function describes the search space and from this the best solutions can be selected. As stated earlier GAs are suited to problems with unknown search spaces and finding a suitable fitness function can be problematic.

GAs can be distinguished from other search methods by its operators.

The three most common operators are selection, crossover and mutation. The most basic method is known as the roulette wheel method. This gives parents with a higher fitness value a greater chance of selection to produce children.

$$\text{Probability of individual selection} = \frac{\text{individual fitness}}{\text{total population fitness}}$$

This method of selection for reproduction can be paralleled with Darwin's survival of the fittest. Parent strings are crossed over to produce 'children'. There are several methods of crossover the simplest being one point crossover.

Parents	Children	
00110|10	0011001	
10011|01	1001110	| denotes the crossover point.

To enable schemata to be preserved, especially in longer more practical strings, two point crossover was introduced. Bold type indicates schemata and | crossover points.

Parents	Children
1001|1101|**01**10	**1001101101**10
1100|**1011**|0111	110011010111

Sometimes it is not possible to preserve the schemata with two point crossover. Uniform crossover can be used and a template placed over the strings to determine the crossover points.

Parents	Template	Children
01001101110	11011000111	00001101010
10010111010		11010111110

The most suitable crossover method is problem dependent. It is also closely related the type of coding used for the chromosomes as some crossover methods can produce illegal children or eliminate strong schema.

The other common GA operator is mutation. This is usually activated only at a very low rate, e.g. one per thousand. Mutation creates variation, can move GAs out of sub-optima and prevent premature convergence. Theoretically at any time during a GA search any point of the search space can be reached via mutation, this has given rise to mutation being referred to as the GA insurance policy. Mutation is also present in nature.

A highly fit individual can eliminate all other weaker strings in a few generations. If all the strings have very close fitness values, the fittest do not have a much greater chance of proceeding to the next generation. To alleviate these problems the fitness values can be scaled by linear normalisation. This is done by ranking all the strings in order of their fitness values. The fittest string is given a predetermined value and all other strings are then given uniformly decreasing values according to their rank. This has the effect of spreading out the values and hence giving the fittest strings the best chance of survival, but not eliminating the possibility of other strings reproducing.

While a highly fit individual can be dominant, it is also feasible that it may be eliminated due to the probabilistic nature of selection. It is possible to preserve the fittest string(s) for reproduction by use of the elitism operator, to ensure that the current best solutions survive.

There is great potential for using GAs in optimisation and as yet there has been relatively little work or hype about this area (anonymous, 1993). There are many papers available in conference proceedings such as The International Conference on Genetic Algorithms held every two years since 1985 or the Congress on Evolutionary Computation. There are journals published in this area including Evolutionary Computation and IEEE Transactions on Evolutionary Computation.

3.3 HYBRID GENETIC ALGORITHMS

Pure GAs are a blind search technique which means that they are good at solving many problems, but it is this generality that counts against them when solving specific problems. Adapting GAs to suit a problem or incorporating domain knowledge can successfully overcome this, but domain knowledge is often not available. Combining two or more techniques to obtain the best facets of each is know as hybridisation. There are many examples of hybridised GAs being used to successfully solve problems e.g. (Davis, 1987, Wienholt, 1993). A large amount of work has been carried out combining GAs and neural networks, for examples see (Thierens *et al*, 1993), (Alba *et al*, 1993) or (Bishop *et al*, 1993). GAs have even been used to improve GAs (Friesleben and Härtfelder, 1993). Many other methods such as tabu search and simulated annealing, have been hybridised, examples of hybridised GAs can be found in (Winter *et al* 1995) and (Renders and Bersini, 1994).

3.3.1 Pareto Optimality GAs

Pareto optimality GAs are a form of multi-objective GAs. A definition of Pareto Optimality is given in Mason et al (1998): "The solution to a multi-objective problem is, as a rule, not a particular value, but a set of values of decision variables such that, for each element set, none of the objective functions can be further increased without a decrease of some of the remaining objective functions (every such value of a decision variable is referred to as Pareto-optimal)." or more simply defined in Principia Cybernetica Web (2000) as : the "best that can be achieved without disadvantaging at least one group."

There are several different approaches to combining Pareto optimality into a GA. The GA can be initially run conventionally to produce a generation of individuals. The Pareto optimality element then determines a set of dominant values from this population. A decision then needs to made as to how to carry these 'superior' individuals forward into the next generation (Sait and Youssef, 1999). The 'Pareto individuals' could be designated a higher probability of entering the mating pool, or by the use of an elitism operator. A variant of binary tournament selection can be used to incorporate Pareto optimality into a GA (Louis, 1997). Murata and Ishibuchi, (1995) present MOGA which includes Pareto and 'uses a weighted sum of multiple objective functions to combine them into a scalar fitness function' where the weights are randomly specified for each selection operation. An elitism strategy is utilised to retain the best individuals.

(Ishibuchi and Murata, 1996) propose a hybrid algorithm based on MOGA (Ishibuchi and Murata, 1995) in which a local search procedure is applied to each solution generated by the GA and 'the choice of the final solution is left to the decision maker's preference'. If all local neighbours of each GA solution are examined then a large part of the computation time is spent on local search. The proposal is then to only look at 'k' solutions, where 'k' is chosen by hand. The paper concludes that the number of local search carried out had a large effect on the efficiency of the algorithms and the best results were obtained with random real number weights assigned to the criteria.

One advantage of this method is that 'Pareto-optimal selection also eliminates the need to combine disparate criteria into a single fitness criteria as is usual in genetic algorithms (Louis, 1997). Only two criteria are combined in Louis and Rawlins, (1993) since as the number of criteria increases the possible combinations rise combinatorially. A Pareto GA is applied to the design of satellite constellations in Mason et al, (1998), which also comments on the requirement for a priori knowledge in this case. An example of a Pareto-optimal GA for flow shop scheduling with three criteria, makespan, tardiness and total flowtime, is presented in Ishibuchi and Murata (1995). Ishibuchi and Murata (1996) also lists other papers with extension of GAs to multi-objective optimisation. Pareto optimality is examined in relation to genetic programming in Langdon (1995) which concludes 'that smaller evolutionary steps might aid GP in the long run'.

The GA hybrids presented in this thesis also combine global and local search as do the Pareto-optimality GAs. In this thesis the local search element is carried out by EVOP on each of the individual GA solutions. Due to the small populations of the hybrids in this thesis the required additional running time did not pose a problem. Pareto-optimality GAs look to combine many objectives into a single measure and locally search in the most promising direction, then often use elitism to retain the best individuals. The EVOP-GA hybrids presented in this thesis search with a single objective, but could be extended by careful construction of the objective function to include multi-objectives. The approach taken in this thesis echoes the 'keep-it-simple' philosophy of EVOP, with the added, and intended, benefit of fast run-times.

3.4 SOFTWARE SELECTION

3.4.1 EVOP Software

There is little information available on EVOP software. It is possible that some companies using EVOP have written in-house software, but until very recently commercial EVOP software was not available.

Two experimental design software packages that incorporate EVOP have been located, both include Simplex EVOP. MultiSimplex offer the MultiSimplex package running as an 'add on' to Excel version 5 or higher. The Statistics Department at Leeds University developed PEXLAB, PLanning Experiments LABoratory which is a Fortran77 program which was available for evaluation by ftp only.

EVOP as discussed earlier in section 3.1, is a little used technique and it is unsurprising that there appears to be a very small amount of software available. None of the occurrences of EVOP found in the literature referred to using software and some presented results were on hand filled EVOP sheets. MultiSimplex promotes the inclusion of EVOP but the actual software only allows Simplex EVOP. The emergence of QS-9000 as an automobile industry standard which includes EVOP as a process improvement technique could induce more commercial EVOP software packages. EVOP is a simple technique which can be written into a spreadsheet as demonstrated by the new work in section 3.1.7. but commercially available software would lead to greater use of the technique.

3.4.2 GA Software

Artificial intelligence software is a rapidly expanding area with many commercial packages now available. At the point in this study where software was required, GA software was available from commercial sources, such as Evolver from Pallisade which runs in Excel spreadsheets, but these were not very flexible. There was no choice as to the type of GA, e.g. steady state or not. If not an option on the user screen, then changing some parameters could involve delving into the code of the programme, which in itself is not a simple task to draw up from behind the user interface. At the time of selecting suitable software Evolver was a beta release and testing revealed that it was not reliable on Windows 3.x. Many GA packages are freely available on the internet and a large proportion are a result of academic research, such as Mattlib, but they are also often shareware complete with programming bugs. Many have quite demanding minimum requirements for both PC processor and programming knowledge of specific languages. The main drawback to most of these packages was that the software which had most potential for this study consisted of lengthy code which was usually not documented and with sparse commentary in the code. This made de-bugging and altering the code to suit this study a potentially monumental task. The third category of GA software consists mainly of research institute based written software for which there is a nominal charge, but the code is generally more reliable than the freeware and support is also often available. A list of software available at the time of software selection for this study can be found in Heittkötter and Beasley (2000).

3.4.3 Criteria for Selection

GAs can be run extremely quickly on large machines which are highly parallel but this is not what is readily available in industry. GAs must reach an acceptable answer quickly on an average machine for their use in industry to greatly increase. Industry is more likely to use a GA that runs on less expensive, or existing equipment and software. As the thrust of this study was to combine the local and global search capabilities of EVOP and GAs to produce an optimal performance in a relatively short time, this was the main restriction on the choice of software to used for the hybrid GAs proposed for this thesis. The criteria for selection was that the GA could run on a relatively small machine with acceptably priced software and the EVOP element be included by a relatively small amount of programming.

Matlab is a commonly used software system with full documentation, support available from Math Works and discussion groups on the internet. The Matlab GA Toolbox from Sheffield University was selected as it runs on widely used platforms and does not require more than an 'average' PC.

3.5 SUMMARY

Evolutionary Operation (EVOP) and Genetic Algorithms (GAs), the two methods selected in chapter two, have been examined in greater detail so that new forms of optimisers which combine both global and local search may be produced for this study.

A review of the literature gave details of published hybrid GAs, including those featuring Pareto-optimality, combining local and global searching, to place in context the new hybrid methods from this thesis that are based on GAs and EVOP. The availability of software for both EVOP and GAs is then discussed. Due to the lack of EVOP software available, also to give an understanding the programming required for EVOP, AutoEVOP software created for this study is described. Finally, a summary is presented of the process of selection of software for the hybrid methods created and tested in this thesis in the next two chapters.

4. SMALL POPULATIONS AND HYBRIDS

The aim of this chapter is to illustrate the processes undertaken to select suitable operators for the genetic algorithms used in this study and to show any influences these operators have on hybrid GAs in comparison to standard microGAs on a set of test functions. Initial study is devoted to selecting appropriate parameter values for the GAs, such as the coding alphabet and the size of populations to be examined. Further consideration is then given to the classic operators such as selection method, crossover and mutation for these small population GAs. A range of sets of experiments are undertaken to determine the influence of these operators, especially population size and generation gap. These experiments will allow for optimum combination of the global and local search capabilities of the two chosen methods.

4.1. SELECTION OF GA PARAMETERS

4.1.1 Population Size

As the intention of the study was to produce a search method that combined both local and global search, that could be used in on-line situations it was necessary to have a small population GAs to ensure minimal computational time. There appears to be no widely accepted definition of 'small' with respect to genetic algorithm population size. As populations of only a few hundred are regarded small by some, the thirty strings or less per generation which are used for the experiments described here would be widely accepted as a small population. A few published results suggest "that population sizes as small as thirty are quite adequate in many cases. Nevertheless, little has been published that is relevant to *really* small populations" (Reeves, 1993). Arabas et al (1994) states that population size is important, yet "the knowledge about proper selection of GA parameters is still only fragmentary and has rather empirical background". Tests were therefore carried out with populations from as large as thirty to as small as five.

4.1.2 Coding Alphabet

There is still debate as to the best alphabet to use with GAs and good results have been obtained with real numbers and q-ary alphabets. Binary coding was selected for the experimental GAs as it has been shown by Reeves (1993) that the minimum population size for binary coded strings is much less than for q-ary alphabets. Krishnakumar (1989) also concluded that binary is the best form of coding with small populations. Binary coding also means that the EVOP element of the hybrid GAs discussed in chapter three can be easily incorporated, since the step size is restricted to the inversion of a single binary number and not an arbitrarily selected step size as for a real or q-ary number.

4.1.3 GA and Experimental Design Hybrids

"Simply taking a small population size and letting them converge is certainly not useful" (Krishnakumar, 1989). The similarities and differences between experimental design and GAs are explored in Reeves and Wright (1995) which concludes that GAs and the experimental design method studied could both potentially benefit from combining certain aspects of each of the methods. There is literature such as Davis (1993), Renders and Bersini (1994) and Zalzala and Fleming (1997), which discuss combining GAs with hill climbing methods. EVOP is a little used method of experimental design which could be compared to a single stepping multi-directional hill climber and could provide an efficient constituent of a hybrid GA.

4.1.4 EVOP and GA Hybrid Method

The GA element of the hybrids in this study were created as M *x* N matrices, where M represents the length of the chromosomes and N the size of the population. The format of a chromosome is similar to those described in section 3.2. After an evaluation of the child population the EVOP element is added by multiplication of the GA matrix with a suitably sized binary matrix, with zeros and ones placed to create the effect of EVOP on the GA chromosomes. If the chromosome is split into many sections representing different elements, the EVOP matrix must be created to change each section. The effect of 'stepping' in different directions and then moving in the best direction is achieved by multiplying the GA matrix with various forms of the EVOP matrix and retaining the best solution with an elitism operator.

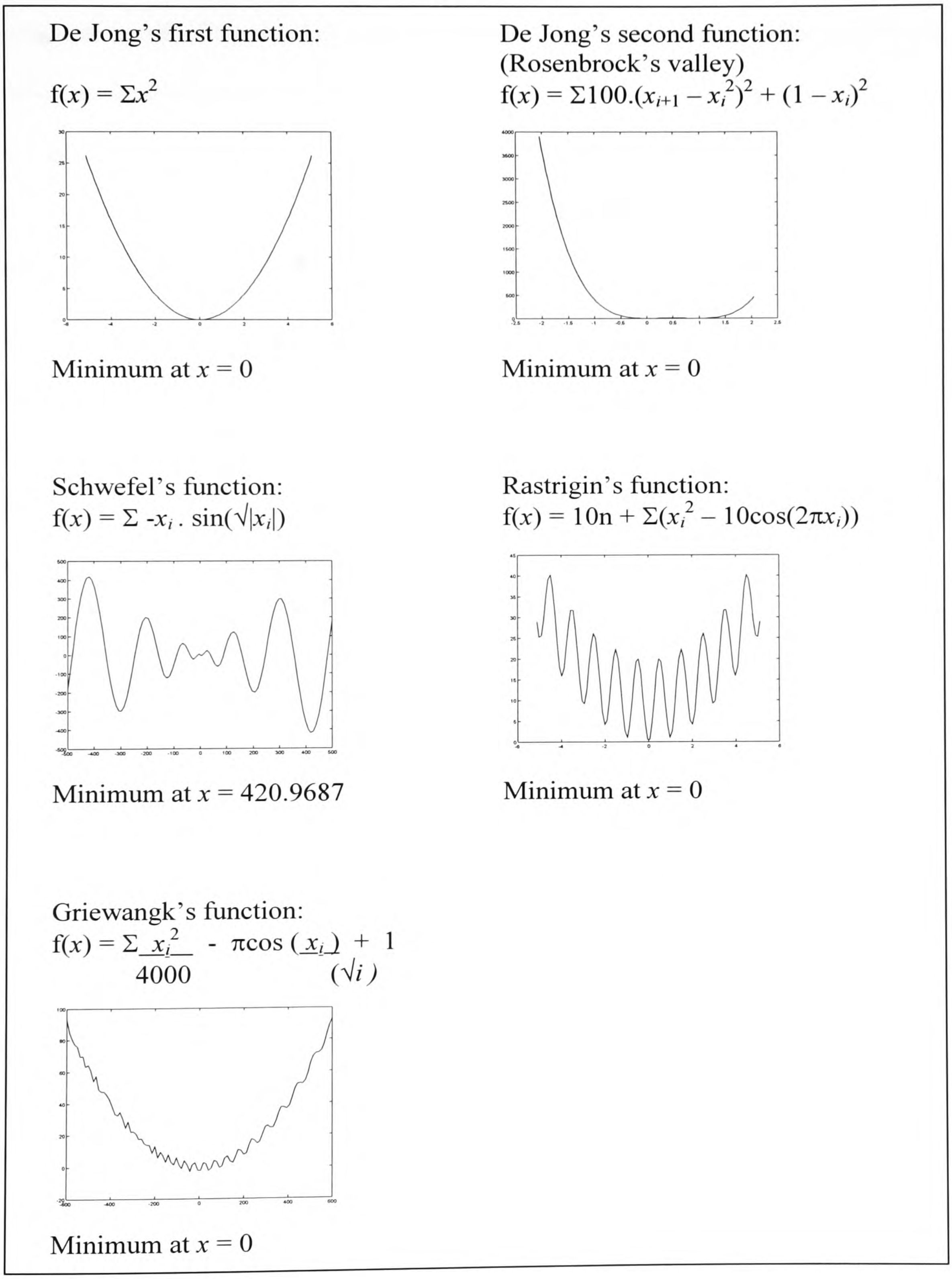
4.1.5 Test Functions

The functions used to test the GAs in this study were chosen as they often appear in the literature as a standard test functions. Published results appear for larger population and generation GAs than those tested here, but using standard functions allows for comparison with existing and future works. The set of functions include De Jong's first and second functions from De Jong's traditional suite of test functions, Schwefel's, Rastrigin and Griewangk's functions. This suite consists of five functions which vary greatly from the relatively simple first function to the more complex spaces of the later functions. The functions used in this study are described in figure 4.1 by their mathematical functions and illustrated by two dimensional Matlab plots.

4.2 SELECTION OF GA OPERATORS

4.2.1 Population Selection

A selection method suitable for small population GAs was required. There is very little literature available on GAs with populations as small as those proposed. Roulette wheel selection (RWS) is a common method where the probability of a string being selected for inclusion in the mating pool is proportional to its fitness. With very small populations, although there is always the possibility of any individual being selected, there is a risk of domination by one or a few individuals



De Jong's first function:

$f(x) = \Sigma x^2$

Minimum at $x = 0$

De Jong's second function:
(Rosenbrock's valley)
$f(x) = \Sigma 100.(x_{i+1} - x_i^2)^2 + (1 - x_i)^2$

Minimum at $x = 0$

Schwefel's function:
$f(x) = \Sigma -x_i . \sin(\sqrt{|x_i|})$

Minimum at $x = 420.9687$

Rastrigin's function:
$f(x) = 10n + \Sigma(x_i^2 - 10\cos(2\pi x_i))$

Minimum at $x = 0$

Griewangk's function:
$f(x) = \Sigma \frac{x_i^2}{4000} - \pi\cos\left(\frac{x_i}{(\sqrt{i})}\right) + 1$

Minimum at $x = 0$

Figure 4.1 Test Functions

which are much fitter leading to early convergence, due to the smaller number of 'choices' on the roulette wheel.

Stochastic universal sampling is a suitable technique for selection as it has zero bias due to individuals being selected entirely on their position in the population. Stochastic Universal Sampling (SUS) is sometimes known as systematic sampling.

The initial sample from a population is chosen at random. In some instances this may be restricted to the first X members of a sample, where X represents one n^{th} of the population and n is the number of population members required for the sample. The distance to the next member of the population to be included in the sample is randomly selected. Again restrictions can be applied to this scalar if a certain frequency of sampling is required. The distance to the next and all subsequent sample members is the same as the distance between the first and second members of the sample.

The program code for stochastic universal sampling and roulette wheel selection is very compact. The SUS code contains one more line of executable code than the roulette wheel method code, so the time needed to execute the GA program is not greatly affected by this choice.

Testing was carried out on the standard small population GAs to ascertain the most suitable method. "Standard small population GAs are as defined earlier in section 4.1.1. Tests were run on micro-GAs with populations of five, ten and twenty. Preliminary testing showed that 'best' solutions were reached usually within the first twenty generations. Therefore the GAs were stopped after thirty generations. Each test was repeated at least one hundred times. For the results shown in figure 4.2 the test function used was De Jong's first function where the known solution is zero. Each generation completely replaced the previous one. Crossover was single point. Figures 4.2 to 4.4 each show three separate sets of one hundred runs at the above settings.

For all population sizes the mean output of both RWS and SUS were similar. As the population size increases the SUS is more clearly shown as a consistently better method. A similar pattern emerged for the maximum output, i.e. worst case, output. Although a more mixed set of results were obtained for the minimum output achieved, it can be clearly seen that for a population of twenty SUS gives optimum results on a par with those obtained by RWS.

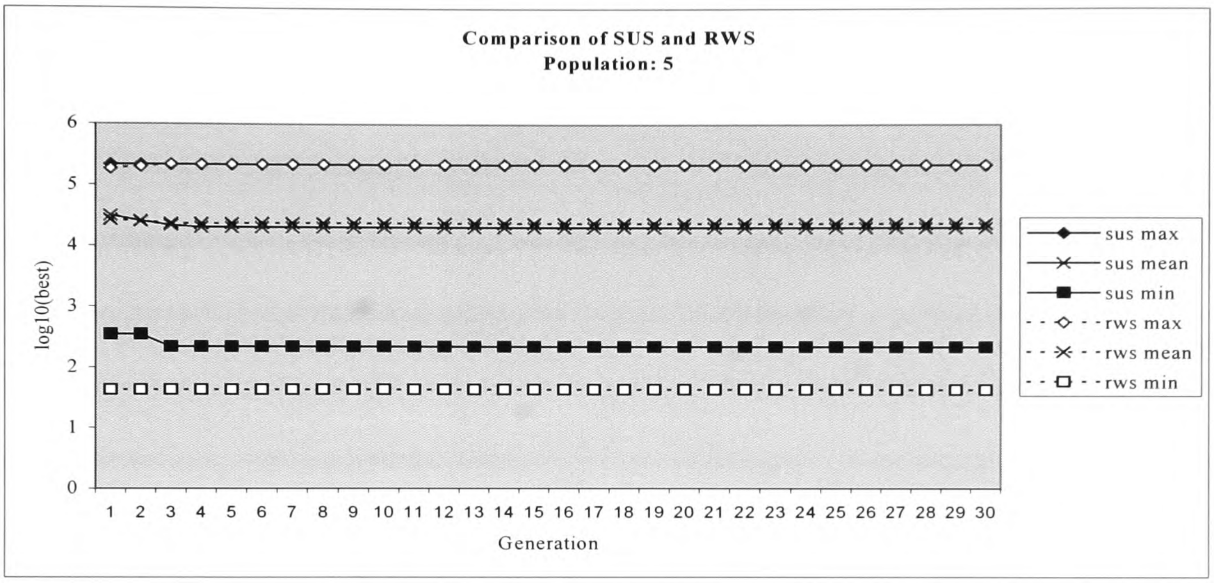


Figure 4.2. Sample comparison of Stochastic Universal Sampling and Roulette Wheel Selection for a micro-GA with a population of five.

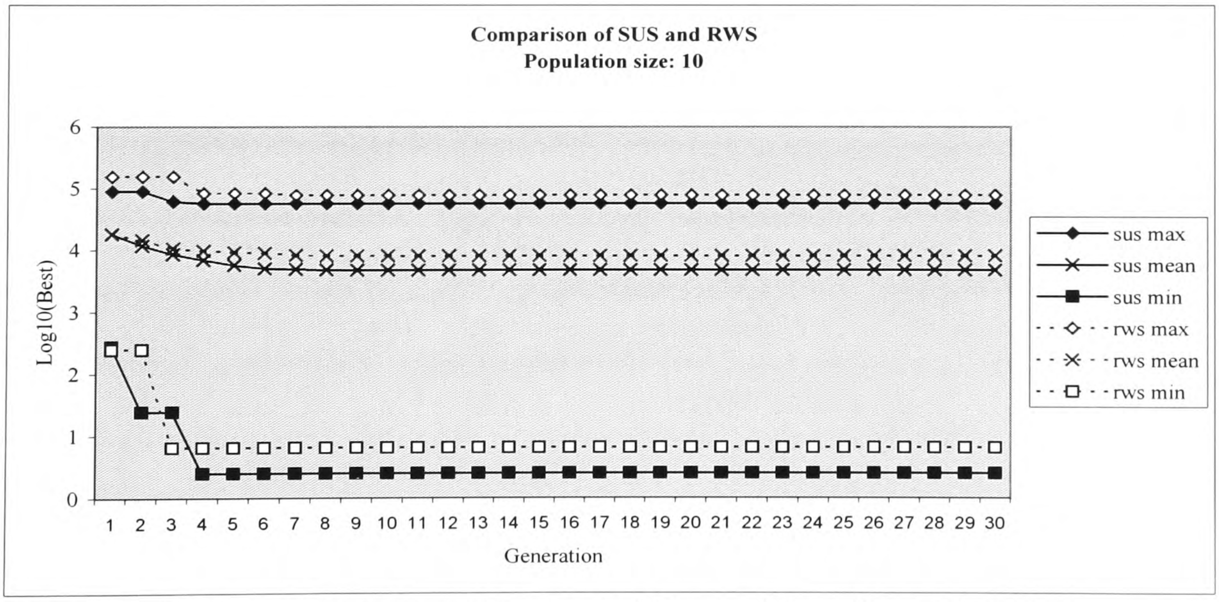


Figure 4.3. Sample comparison of Stochastic Universal Sampling and Roulette Wheel Selection for a micro-GA with a population of ten.

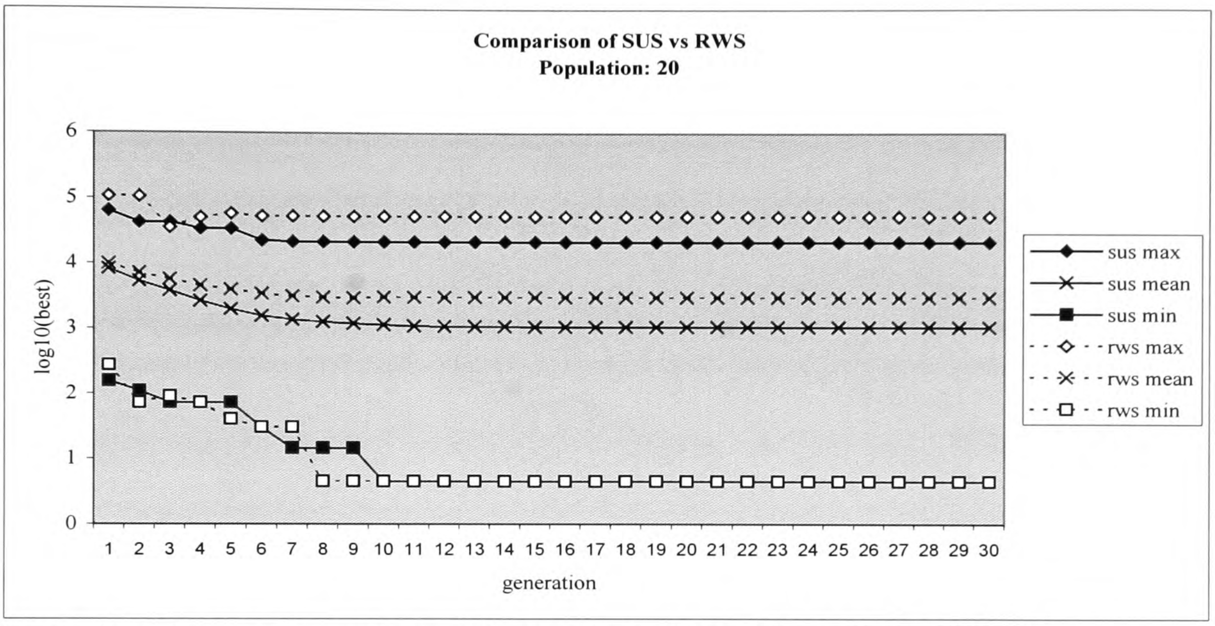


Figure 4.4. Sample comparison of Stochastic Universal Sampling and Roulette Wheel Selection for a micro-GA with a population of twenty.

Collation of the results of these tests also revealed that SUS is a more robust method than RWS in that in the majority of cases it was found that there was less variation between the best and worst solutions using the SUS selection method. These tests were repeated as many as thousand times in sets of one hundred and although the actual values obtained varied slightly the general trends were identical. A sample of summarised results is shown in tables 4.1, 4.2 and 4.3.

As a result of these tests and other sample experimentation with large population GAs, SUS was selected as the sampling method for use with future experimental GAs.

Generation	SUS			RWS		
	Max	Mean	Min	Max	Mean	Min
5	5.354116	4.211948	1.563142	5.342308	4.521769	3.22004
10	5.354116	4.211051	1.485148	5.342308	4.516849	3.22004
15	5.354116	4.211051	1.485148	5.342308	4.516849	3.22004
20	5.354116	4.211051	1.485148	5.342308	4.516849	3.22004
25	5.354116	4.211051	1.485148	5.342308	4.516849	3.22004
30	5.354116	4.211051	1.485148	5.342308	4.516849	3.22004

Table 4.1 Comparison of a sample of results for a population size of five.

Generation	SUS			RWS		
	Max	Mean	Min	Max	Mean	Min
5	4.760520	3.758525	0.398789	4.924986	3.968418	0.813762
10	4.757428	3.678901	0.398789	4.894780	3.911647	0.813762
15	4.757428	3.678497	0.398789	4.894780	3.908215	0.813762
20	4.757428	3.678497	0.398789	4.894780	3.908215	0.813762
25	4.757428	3.678497	0.398789	4.894780	3.908215	0.813762
30	4.757428	3.678497	0.398789	4.894780	3.908215	0.813762

Table 4.2 Comparison of a sample of selected results for a population size of ten.

Generation	SUS			RWS		
	Max	Mean	Min	Max	Mean	Min
5	4.52954	3.295254	1.861187	4.760384	3.596094	1.608304
10	4.332893	3.057405	0.654061	4.719604	3.481652	0.654061
15	4.332893	3.02555	0.654061	4.719604	3.477431	0.654061
20	4.332893	3.025082	0.654061	4.719604	3.478151	0.654061
25	4.332893	3.025027	0.654061	4.719604	3.478151	0.654061
30	4.332893	3.025027	0.654061	4.719604	3.478151	0.654061

Table 4.3 Comparison of a sample of selected results for a population size of twenty

4.2.2 Crossover Operator

Single point crossover was initially chosen for testing with relatively short strings. Since the overall aim of is to produce a fast GA the approach to programming was to use programs with concise code to enable faster processing times. Single point crossover being the simplest option also corresponded with the general philosophy of the testing to keep the search algorithm as streamlined as possible. A search of literature revealed no clearly superior crossover method for small population GAs.

Although multi-point crossover has the potential to cover more of the search space, in conventional GAs it does not generally lead to early convergence (Sait and Youssef, 1999). As a rough guide to speed of operation, the code for single point crossover is three lines of code to be compiled compared to forty four lines for multi-point crossover with the software used for testing. As a fast GA was the objective, the crossover method with the least code to be compiled was selected, so multi-point crossover was excluded from the initial experiments.

4.2.3 Mutation Operator

The third classic GA operator, mutation, was turned off for these experiments. To have an effect with small populations the mutation rate would need to be greatly increased from the usual rate of one in thousands to one in hundreds or tens. However, at this level it could have the effect of turning the search into a random walk. As described in section 3.2.2.2, mutation is often used as a GA 'insurance policy' so that at any stage of the search it is possible to reach the entire search space. With small populations, mutation has little influence, but with the new experimental GAs no area of the search space is excluded due to the EVOP element.

4.2.4 Generation Gap

Generation gap determines the proportion of population to be reproduced. For example, with a population of 100 and a generation gap of 0.7, once selection and crossover have taken place there will be 70 new individuals in the population of 100 in the new generation. With the software used it is possible to have an increasing population with this operator, but this option was not utilised to restrict the execution time of the GA.

It was noted that steady-state GAs have faster processing times than conventional GAs as relatively few new strings need to be stored at each generation. Due to this potential time saving the influence of the generation gap was extensively tested on the new GAs and the standard small population GAs. De Jong's first test function was used as the objective function for the initial tests as the ideal solution is known and the function is relatively simple. The tests on generation gap will be discussed in sections 4.2.4 and 5.3.1.

4.3 HYBRID TESTING

4.3.1 Initial Migration and Robustness Testing

Migration, or the Island model (Georges-Schleuter, 1992) divides a single population into a number of sub-populations. Each sub-population behaves as a 'normal' GA but periodically individuals move or migrate from one sub-population to another. Migration can increase diversity in GAs and has been shown to improve results in some cases (Mühlenbein et al, 1991), (Starkweather et al, 1990). Small population GAs are prone to stagnation due to the low number of strings available for breeding. It was decided to examine if the benefits obtained in the literature could be achieved for very small populations and a comprehensive set of experiments was conducted.

For these experiments robustness of the algorithm is defined as the range of solutions found by the hybrid over fifty runs at identical initial experimental settings. Figure 4.5 shows the influence of the migration rate on finding a good solution and the robustness of the settings. Figure 4.5 was generated using De Jong's first test function in two dimensions and with the generation gap set so that each generation completely replaces the last one. Clearly the best solutions are found at 50% migration rate, but the robustness of the hybrid is best at a migration rate of 30%. A comparative study was carried out using a small population standard GA. For all cases the best solutions found by the hybrid were better than those found by the standard GA.

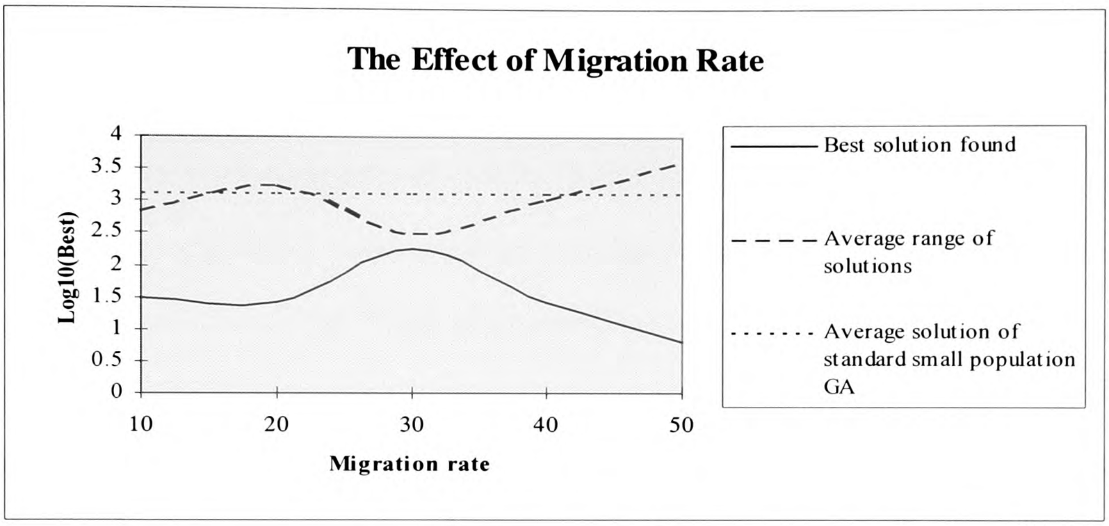


Figure 4.5. The effect of migration rate on the solution found.

A further investigation was carried out at migration rates of 30% and 50% as these gave the worst and best solutions respectively. Results of this investigation are shown in Figure 4.6. The diagram shows the effect of the generation gap at migration rates of both 30% and 50% and the range of solutions found. There is an almost direct inverse correlation between robustness and finding an optimal solution at both migration rates. At a migration rate of 30% with a generation gap of one the hybrid is robust, but the solutions found are not optimal, yet the better solutions are found with a generation gap of between 0.6 and 0.8 where the hybrid is least robust. At a migration rate of 50% the value of the best solution found oscillates as the generation gap increases. At all settings for generation gap at both migration rates tested the best solutions found by the hybrid are an improvement on those found by the standard small population GA.

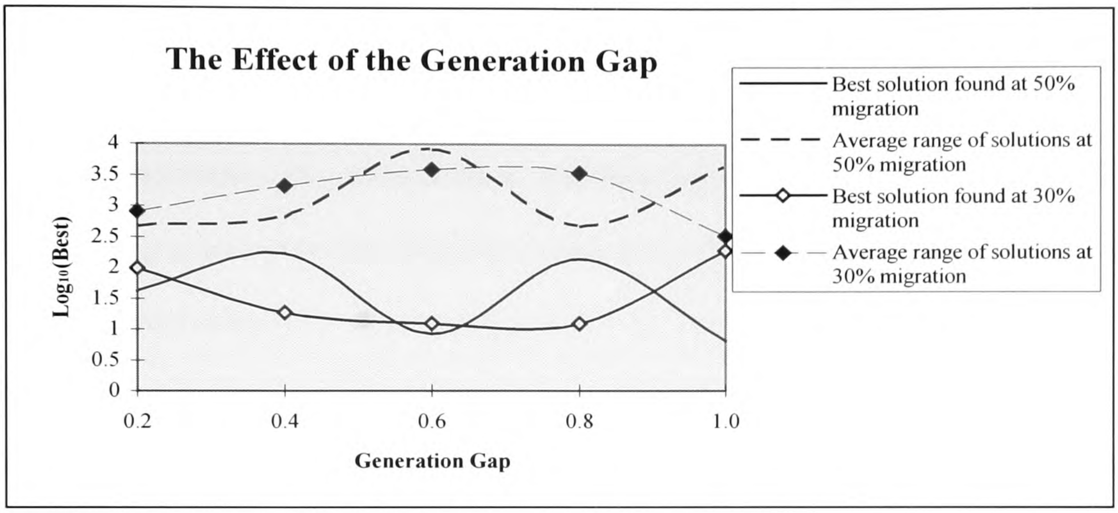


Figure 4.6. The effect of generation gap on the solutions found.

4.3.2 The Effect of Generation Gap

If 'breeding' of the population results in fewer new individuals than in the original population, then the fractional difference between the new and old populations is known as the generation gap (DeJong and Sarma, 1993).

Following the initial testing with migration that included some study of the generation gap, a more comprehensive set of tests were carried in order to identify effects due to the setting of the generation gap. For all experiments each GA, standard or hybrid, was executed one hundred times. Although the software used for the experiments randomly generates the first population, the random number generator used in the code for the programs is not truly random, but pseudo random numbers between zero and one are generated in a set order. To overcome any bias that this may introduce into the results the entire set of experiments were listed and assigned a random order in which to be run, this meant that no two runs of a GA experiment with the same settings would have the same starting points. Perhaps more importantly, the comparative standard small population GA were run in exactly the same order, thus they were provided with the exactly same starting points as the hybrid GAs.

4.3.2.1 Generation gap with a population of twenty

The first set of collated results are for a population size of 20 chromosomes. The main investigation was to survey any effect of the generation gap, but for each experimental configuration the solutions found at generation 5, 10, 15, 20 and 30 were recorded. Although widely accepted that if a GA is allowed to run to convergence over many generations good solutions are found, the literature did not indicate the differences found between relatively small generation numbers.

4.3.2.1.1 De Jong's first function

Initially the GAs were tested on De Jong's first function. It was found for both the standard microGA and the hybrid GA that after thirty generations the recorded results show that there is a general trend of greater improvements to the initial population as the generation gap increases for a population of 20 chromosomes. A summary of these results is shown in tables 4.4 and 4.5. This general trend of improvement was also found for De Jong's second function, see the next section, 4.3.2.1.2.

The following tables of experimental data, e.g. tables 4.4 and 4.5, show the summarised comparative performance of the GAs under test. For each setting of the GA parameters the experiment was run one hundred times, each run starting at a random population as explained in section 4.3.2. The solutions found at the end point of the GA run, and in some subsequent tables at certain generations of the search, were compared with those generated for the initial population. This demonstrates if the GAs are producing improvements in the solutions found and allows comparison of the effects of the different settings.

The left-hand side of the table shows how many of the one hundred runs produced final solutions that were worse, showed no change or an improvement on the initial random population. The right-hand side of the table details how great the improvements were of those solutions on the left-hand side of the table that proved to be better than the initial population. This quantification gives a clearer indication of the amount improvement gained by particular settings. The three categories that are listed on the right-hand side of the tables are '30% better', '50% better' and '70% better'. Percentages are used to allow comparison between the results produced on different test functions, where the range of values of *x* differ, see figure 4.1 and section 4.1.5. The layout of tables of experimental data described above is applicable to all tables from 4.4 to 4.16.

Generation	No. of GAs compared to initial population					
Gap	Worse	No change	Better	30%better	50%better	70%better
0.1	47	6	47	2	1	0
0.2	20	9	71	6	1	1
0.3	21	4	75	12	3	1
0.4	14	2	84	12	4	2
0.5	7	4	89	25	5	2
0.6	4	0	96	20	4	2
0.7	3	4	93	22	12	4
0.8	4	3	93	27	12	3
0.9	5	1	94	25	10	4
1.0	1	1	98	24	8	6

Table 4.4. Performance of a standard small population GA on De Jong's first function at the thirtieth generation with a population of 20 individuals.

Generation	No. of GAs compared to initial population					
Gap	Worse	No change	Better	30%better	50%better	70%better
0.1	41	1	58	3	0	0
0.2	20	1	79	11	5	2
0.3	22	2	76	11	1	0
0.4	17	4	79	15	7	2
0.5	9	2	89	22	8	2
0.6	11	0	89	19	4	1
0.7	7	1	92	25	11	6
0.8	7	3	90	25	5	0
0.9	3	1	96	22	8	1
1.0	1	1	98	29	11	2

Table 4.5. Performance of a hybrid GA with a population of 20 on De Jong's first function at the thirtieth generation.

The results were filtered to show the chromosomes that showed the most improvement (any improvement, greater than 50% and greater than 70%) over the initial population for all test functions. The greatest improvement for both test functions was at a generation gap setting of 0.7, especially for the hybrid. This could also be seen across the generations with the results sampled at generation 10, 15 and 20.

Regression analysis was used to study the relationship between the performance of the hybrids and the generation gap. Regression analysis of the results with at least a 50% improvement after thirty generations showed that the visual trend could be described as follows:

Standard Small Population GA:	$y = 1.2x - 0.5$
Hybrid:	$y = 0.9x + 1.1$

where x is generation gap and y is percentage improvement on initial population, which gives:

Standard Small Population GA:	$R^2 = 71\%$
Hybrid:	$R^2 = 53\%$

R^2 is the value derived by standard regression analysis and shows the strength of the relationship between the variables in the equations, in this case percentage improvement and generation gap. The calculated values show that for the standard small population GA, 71% of the improvement is due to the generation gap. The influence is not as strong on the hybrid GA, but this is expected as there is a greater random element due to the use of EVOP and the 'peak' at a generation gap of 0.7. Visually, there appeared to be little improvement between the recorded results after thirty generations compared to those recorded after twenty generations. Regression analysis of the results obtained after twenty generations gave the following:

Standard Small Population GA:	$R^2 = 70\%$
Hybrid:	$R^2 = 60\%$

Analysing the difference between the results obtained after thirty generations compared to those for twenty generations the change was less than 0.002%. The analysis suggests that extrapolation, i.e. increasing the generation gap, would give better results but that of course is not possible as a generation gap cannot be greater than the original population, with these steady state GAs. However, it underlines the general concept of the more generations a GA has the better the answer.

For a standard small population GA, the generation gap clearly has an influence. Although the regression value obtained did not shown perfect correlation, a positive relationship was shown despite the naturally random nature of GAs. The effect of the generation gap is less apparent with the hybrid GA but this was expected due to the nature of the EVOP element.

4.3.2.1.2 De Jong's second function

Once again there was a distinct trend of improvement in the results obtained as the generation gap increased. The results shown in tables 4.6 and 4.7, exhibit a much greater improvement than that seen during testing on De Jong's first function.

Generation	**No. of GAs compared to initial population**					
Gap	**Worse**	**No change**	**Better**	**30%better**	**50%better**	**70%better**
0.1	31	12	57	48	41	34
0.2	34	6	60	52	48	40
0.3	35	3	62	65	58	50
0.4	19	4	77	64	57	46
0.5	10	1	89	69	63	57
0.6	10	2	88	63	56	51
0.7	8	1	91	72	65	60
0.8	5	4	91	72	67	64
0.9	7	4	89	65	61	55
1.0	1	3	96	73	71	68

Table 4.6 Performance of a standard small population GA on De Jong's second function at the thirtieth generation with a population of 20 individuals.

Generation	**No. of GAs compared to initial population**					
Gap	**Worse**	**No change**	**Better**	**30%better**	**50%better**	**70%better**
0.1	35	2	63	48	42	35
0.2	37	0	63	50	45	38
0.3	36	0	64	71	60	53
0.4	19	1	80	63	56	49
0.5	7	1	92	65	61	56
0.6	16	2	82	65	59	52
0.7	17	0	83	69	63	59
0.8	8	3	89	73	64	63
0.9	5	2	93	62	55	51
1.0	1	1	98	72	71	67

Table 4.7 Performance of a hybrid GA on De Jong's second function at the thirtieth generation with a population of 20 individuals.

Calculated in exactly the same manner as section 4.3.2.1.1, regression analysis of the results with at least a 50% improvement after thirty generations showed that the visual trend could be described as follows:

Standard Small Population GA:	$y = 2.6x + 44.6$
Hybrid:	$y = 2.2x + 45.3$

where x is generation gap and y is percentage improvement on initial population, which gives

Standard Small Population GA:	$R^2 = 75\%$
Hybrid:	$R^2 = 61\%$

This shows that for the standard small population GA 75% and for the hybrid 61% of the improvement is due to the generation gap. The analysis was also completed for the results obtained after twenty generations, which gave the following coefficients of determination:

Standard Small Population GA:	$R^2 = 69\%$
Hybrid:	$R^2 = 60\%$

These coefficients both show a correlation between improved results and the generation gap.

4.3.2.1.3 Other functions

Due to the small improvements gained between the twentieth and thirtieth generations further results were obtained at twenty generations, then analysed. Both the standard small population GA and the hybrid GA were tested on the other three functions in the test suite: Rastrigin's, Schwefel's and Griewangk's functions, see section 4.1.5. The results of these tests did not show a similar trend to that identified for the two traditional De Jong functions. The hybrid tested with Schwefel's function demonstrated that the generation gap had little influence on the results. For both Rastrigin's and Griewangk's functions analysis shows that the generation gap does not have a much influence on the results. The standard small population GA showed little correlation with Schwefel's function and very weak correlation for the other functions, which again demonstrates that the generation gap has little influence on the results obtained. Results of the tests using the standard small population GA, population of 20, at the twentieth generation are shown in tables 4.8, 4.9 and 4.10.

Generation	No. of GAs compared to initial population					
Gap	Worse	No change	Better	30%better	50%better	70%better
0.1	21	9	70	2	0	0
0.2	24	7	69	1	0	0
0.3	28	15	57	0	0	0
0.4	34	5	61	0	0	0
0.5	43	2	55	0	0	0
0.6	43	3	54	0	0	0
0.7	45	3	52	0	0	0
0.8	46	4	50	0	0	0
0.9	45	5	50	0	0	0
1.0	49	1	50	0	0	0

Table 4.8. Results obtained on Schwefel's function.

Generation	No. of GAs compared to initial population					
Gap	Worse	No change	Better	30%better	50%better	70%better
0.1	19	8	73	7	5	2
0.2	18	4	78	8	1	0
0.3	8	15	77	6	3	0
0.4	9	8	83	9	4	1
0.5	5	10	85	13	2	1
0.6	3	8	89	20	10	2
0.7	1	6	93	14	8	4
0.8	1	5	94	21	11	3
0.9	1	3	96	26	11	3
1.0	0	0	100	27	13	7

Table 4.9. Results obtained on Rastrigin's function.

Generation	No. of GAs compared to initial population					
Gap	Worse	No change	Better	30%better	50%better	70%better
0.1	11	10	79	11	3	2
0.2	22	9	69	10	8	4
0.3	12	14	74	14	10	9
0.4	12	8	80	9	3	3
0.5	2	8	90	18	11	7
0.6	5	6	89	20	13	11
0.7	2	8	90	22	13	10
0.8	2	5	93	26	18	14
0.9	1	7	92	16	12	7
1.0	1	7	92	26	20	13

Table 4.10. Results obtained on Griewangk's function.

4.3.2.2 The influence of the generation gap

These results lead to the conclusion that the influence of the generation gap depends on the search space so, no single setting of the generation gap can be recommended as superior for all cases. As the best solutions were found with higher generation gap settings for some functions and for other functions high generation gap settings gave comparably good solutions with no degradation in performance as the setting increased, a higher generation gap would be recommended in all cases. The peaks in performance found at 0.7 for the hybrid indicate that this setting would be recommended as a starting point for an empirical search for small population GAs and their hybrids. Large generation gaps are also recommended for standard population GAs (De Jong and Sarma, 1993).

4.3.3 The Effect of a Small Number of Generations

Traditional GAs have many hundred or thousand generations and most published works relate to these type of GAs. Some work has been published on standard small population GAs, sometimes referred to as microGAs (Krishnakumar, 1989), as discussed earlier in Section 4.1 where the view that a population of thirty individuals is a small population was stated. Literature stating the effects of very few generations was limited, so an investigation into the effect of a small number of generations was undertaken. Initial testing was carried out using De Jong's first function using both a standard small population GA and a hybrid.

4.3.3.1 De Jong's first function

With a population of twenty individuals the most change is seen in the first ten generations. After five generations there are often a few individuals which give worse solutions than the original population, with the hybrid GA finding only marginally fewer worse solutions compared to the standard small population GA. As expected the standard small population GA had more individuals that recorded no change from the initial population than the hybrid. This is more noticeable with fewer generations, i.e. five or ten generations.

As stated in section 4.3.2.1.1, results were collated and filtered to show how many individuals had improved, by 30% or more, 50% or more and 70% or more than the initial population. This obviously gave many more results to be assessed but a clearer picture of the GA performance was obtained. Trials were run at all generation gaps but the sample results in tables 4.11 and 4.12 were obtained at a generation gap 0.7. Although both types of GA recorded similar numbers of improved individuals, the hybrid showed more individuals with a greater improvement.

	No. of GAs compared to initial population					
Generation	**Worse**	**No change**	**Better**	**30%better**	**50%better**	**70%better**
5	7	11	82	10	6	1
10	3	5	92	19	11	3
15	3	4	93	22	11	3
20	3	4	93	22	12	4
30	3	4	93	22	12	4

Table 4.11. Performance of a standard small population GA with a population of 20 on De Jong's first function at a generation gap of 0.7.

	No. of GAs compared to initial population					
Generation	**Worse**	**No change**	**Better**	**30%better**	**50%better**	**70%better**
5	16	2	82	13	2	1
10	9	1	90	22	8	3
15	7	1	92	25	10	4
20	7	1	92	25	11	6
30	7	1	92	25	11	6

Table 4.12. Performance of a hybrid GA with a population of 20 on De Jong's first function at a generation gap of 0.7.

4.3.3.2 De Jong's second function

The general trend with different numbers of generations were as expected in almost all cases, especially with fewer generations as illustrated in tables 4.13 and 4.14. The hybrid gave a larger number of results which were actually worse than the original population. The hybrid also produced fewer individuals which had not changed from the initial population. Unfortunately, there was no mechanism to check if the individuals were unchanged or if the individuals were the result of breeding. This showed that the populations of the hybrid were more mobile than those of the standard small population GA: Generations subsequent to the initial population contained more new individuals in the hybrid than in the standard small population GA. Generally the fewer generations, the greater the difference in performance between the hybrid and the standard small population GA, although there was an underlying trend of an improvement in performance as the number of generations increased. However, the improvement between generations was less between generation 20 and 30 and the majority of GAs found good solutions within ten generations. Sample results are shown in tables 4.13 and 4.14.

	No. of GAs compared to initial population					
Generation	**Worse**	**No change**	**Better**	**30%better**	**50%better**	**70%better**
5	13	11	76	62	46	44
10	11	1	88	67	60	57
15	8	1	91	72	65	59
20	8	1	91	72	65	60
30	8	1	91	72	65	60

Table 4.13. Performance of a standard GA with a population of 20 on De Jong's second function at a generation gap of 0.7.

	No. of GAs compared to initial population					
Generation	**Worse**	**No change**	**Better**	**30%better**	**50%better**	**70%better**
5	21	1	78	62	53	50
10	16	0	84	65	59	55
15	17	0	83	69	63	58
20	17	0	83	69	63	59
30	17	0	83	69	63	59

Table 4.14. Performance of a hybrid GA with a population of 20 on De Jong's second function at a generation gap of 0.7.

4.3.3.3 Comparison at ten generations

As the performance of GAs with only five generations gave some erratic results, it was concluded that the smallest number of generations that gave a consistent good performance was ten. To further investigate the effect of very small numbers of generations the hybrid and standard small population GA were tested across the three functions: Rastrigin, Schwefel and Griewangk. As summary of these results are shown in tables 4.15 and 4.16.

For Rastrigin's function the hybrid consistently proved to have a more mobile population which was manifested by producing more individuals that were worse than the original population, especially for generation five, along with showing fewer static solutions, except for when the generation gap was set at 1.0. The hybrid regularly gave more solutions with the greatest improvement from the initial population. Generally the hybrid gave the best performance on this function with a population of ten.

Schwefel's function again showed that the hybrid population was more mobile with fewer individuals with no change from the initial population and more that were worse, especially within the first five generations. With the generation gap set between 0.1 and 0.2 although the hybrid found more improved solutions, the standard small population GA found the best solutions. Otherwise the trend was for the hybrid to find more improved solutions but neither GA making large improvements to the initial population.

The hybrid population again tended towards greater movement when tested on Griewangk's function. The hybrid occasionally did not find as many of the best solutions as the standard small populations, but the best solutions found by the hybrid were of equal quality to those found by the standard small population GA. Generally, the hybrid found more improved solutions.

The hybrid generally, with a few exceptions, for all functions was more mobile and gave improved results. Indicative sample results are shown in tables 4.15 and 4.16. These sample results were obtained at a generation gap of 0.7, but the study included trials at all generation gap settings.

	No. of GAs compared to initial population					
Function	**Worse**	**No change**	**Better**	**30%better**	**50%better**	**70%better**
Schwefel	45	3	52	0	0	0
Rastrigin	1	6	93	14	8	4
Griewangk	2	8	90	22	13	11

Table 4.15. Performance of a standard small population GA at ten generations.

	No. of GAs compared to initial population					
Function	**Worse**	**No change**	**Better**	**30%better**	**50%better**	**70%better**
Schwefel	37	3	60	0	0	0
Rastrigin	7	2	91	16	10	4
Griewangk	9	3	88	21	13	10

Table 4.16. Performance of a hybrid GA at ten generations.

4.3.4 The Effect of Population Size

With only five generations the hybrid did not perform as well as the standard small population GA. Although it found fewer improved solutions, the solutions found were of equal quality as those found by the standard small population GA. The hybrid was also shown to be more mobile for GAs with ten generations. Generally, the hybrid outperformed the standard small population GA, with an exception at a generation gap of 0.2 and a marginally worse performance at 0.8. With twenty generations the hybrid consistently outperformed the standard small population GA. As expected the hybrid population was more mobile.

The greater the population size the better the results found. With a small number of generations this became more important. Populations of five generally gave much poorer results, due to the lack of time to search a large number of points in the search space. A population of twenty individuals gave better results than populations with ten individuals.

4.4 SUMMARY

Stochastic universal sampling has been shown to be more robust than the traditional roulette wheel selection method, and capable of obtaining comparably good solutions. To maintain fast running of the hybrid the simplest crossover method, single point, was used. Mutation was disabled for these small population GAs as the accepted usual rates of one in thousand(s) would have little effect on small populations running only to a few generations. To increase the mutation rate for it to have an effect would risk turning the search into a random walk. For the hybrid, the EVOP element increased the amount of search space that could be reached by the GA and thus compensating for the mutation rate being turned off. Steady state population sizes were used so as not to increase running time of the GAs.

Migration rate was shown to have a bimodal influence on the performance of the small population GAs. Generation gap was extensively tested and was shown to affect the solutions found: although seemingly dependant on the search space, the higher generation gaps generally showed improved solutions. The influence of the generation gap was not as strong for the hybrid as the standard small population GA. Results indicate that a generation gap of 0.7 is to be recommended as a search starting point for small population GAs and their hybrids, as good solutions were found with this setting on many function spaces. This recommendation fits with De Jong's studies (Goldberg, 1988b) on larger populations that concluded that non-overlapping generations perform better in most off-line optimisation situations, but on-line performance 'is not severely degraded by using smaller generation gaps'. A population size of five gave erratic results, but a population of ten gave much better results with this trend increasing as the population size increased for both the hybrid and the standard small population GA. Solutions found improved as the number of generations increased. Five generations did not give very good results but it was noted that most movement occurred in the first ten generations. A population size of ten is therefore recommended as it gives good results, without the time penalties associated with the larger population size of twenty which was also studied.

This chapter has concentrated on the settings of operators for the genetic algorithms used in this study, investigating the influences these operators on small population GAs and their EVOP hybrids. The combination of the global and local searches of GAs and EVOP has been shown to improve the performance of very small population hybrid GAs. The EVOP element is present in all generations of the hybrids investigated in this chapter, further investigation to examine the effects of the influence of the EVOP element on the hybrid GAs is presented in the next chapter.

5. DYNAMIC HYBRIDS

The hybrid GAs tested in the previous chapter contained an element of EVOP which operated from the first to last generations of the GA and are henceforth referred to as static hybrids. The aim of this chapter is to illustrate the investigations undertaken to examine the effects of the influence of the EVOP element on the hybrid GAs, using various standard test functions. In this chapter EVOP is initiated only for certain specified generations, such that the GA proceeds its characteristics change and these GAs are referred to as dynamic hybrids. These dynamic hybrids are compared to each other, to their comparable static hybrid and standard small population GA. The influences of the global and local search elements of the hybrids will be considered. A study is also undertaken into the influence of the generation gap on these dynamic hybrids.

5.1. MODIFICATION OF MICRO GAs

5.1.1 Selection of Parameters

Each hybrid was run one hundred times, due to the probablistic nature of GAs, to overcome any bias that could be introduced by a small number of trials. Stochastic universal selection is used for all hybrids tested here as it has been shown to be a preferred method of selection for small population GAs as explained in Chapter 4. This set of hybrids were run for twenty generations as the earlier hybrid trials had shown that this setting gave acceptable results. With this small number of generations the run time for the GAs is kept to a minimum. Single point crossover was used as this is the simplest method with the least computational demands and hence running time. A population size of ten was used as this was shown in the previous chapter to give good results.

5.1.2 Control of EVOP Element

As discussed in Chapter 3, EVOP is really a local search method whereas GAs are regarded as global search methods due their parallel search technique. Previous hybrids discussed in Chapter 4, combined EVOP with a standard small population GA to search the problem domain. GAs initially search large areas of the problem space, but methods such as EVOP are better at searching small areas. To capitalise on the strengths of both methods, experiments were conducted with hybrids which varied from pure GA to GAs with elements of local search. The initial expectation was that improved performance would be gained by tuning the hybrid from a pure GA to a GA with EVOP as the search proceeded; the EVOP element being restricted to the later generations of the hybrid. The GA conducts the initial search to find the most promising regions (Renders and Bersini, 1994), which are then more closely investigated by the EVOP element. This can be achieved by inserting a simple variable test in the GA code so that EVOP is only executed when a certain condition is met, for example, "EVOP to be present only after ten generations", provided that a variable has been allocated to count the number of generations.

5.2 DYNAMIC HYBRID TESTING

To determine the influence of the EVOP, several hybrids were created, as shown in table 5.1, with EVOP only present during later generations, initial generations or EVOP initiated with increasing regularity as the generations increased.

Hybrid	**Generations EVOP initiated**
GEVO-1	>15
GEVO-2	>10
GEVO-3	>5 and <16
GEVO-4	2, 6,7, 11,12,13, 17,18,19,20
GEVO-5	5, 10,11, 14,15,16, 18,19,20
GEVO-6	<6
GEVO-7	<11

Table 5.1. Initiation of EVOP in hybrids.

The first two hybrids (GEVO-1 and GEVO-2) were designed to show any effect on the solutions found with EVOP being introduced towards the final generations of the GA. GEVO-3 was designed to show the impact of introducing EVOP only in the middle generations of the search. The effect of gradually introducing increasing amounts of local search was investigated using GEVO-4 and GEVO-5. GEVO-6 and GEVO-7 were to investigate the effect including a local search element only in the early part of the algorithm. All hybrids were tested using the five traditional functions at generation gaps varying from 0.6 to 1.0 as previous experimentation (see section 4.3.2) had indicated that better results could be obtained by using higher generation gaps. This means that this set of experiments consisted of 17,500 trials of various hybrid GAs, which were then compared to standard small population GAs and to the hybrids tested in the previous chapter.

5.2.1 Testing on De Jong's First Function

5.2.1.1 Hybrid One – GEVO-1

The first comparisons were made with a generation gap setting of 0.7. As with the first type of static hybrid (Hybrid 1) this dynamic hybrid GA (GEVO-1) was much more mobile than the standard small population GA (PGA) but not as mobile as the initial static hybrids. GEVO-1 found more improved solutions than the standard small population GA, on a par with those found in the previous chapter. GEVO-1 found the solutions with the greatest improvement on the initial population as shown below in table 5.2. GEVO-1 found approximately the same number of improved solutions as the comparable static hybrid (Hybrid 1), both of which were better than the standard small population GA. GEVO-1 gave fewer solutions that were worse than the initial population, yet found more of the best solutions. The format of tables 5.2 to 5.20 follows the format described in section 4.3.2.1.1.

		No. of GAs compared to initial population						
GA	**Generation**	**Worse**	**No change**	**Better**		**30%better**	**50%better**	**70%better**
GEVO-1	5	7	14	79		2	0	0
Hybrid 1	5	16	3	81		5	1	0
PGA	5	10	20	70		2	0	0
GEVO-1	10	5	9	86		5	2	1
Hybrid 1	10	13	2	85		6	1	0
PGA	10	8	12	80		6	0	0
GEVO-1	15	5	9	86		7	3	1
Hybrid 1	15	12	2	86		6	1	0
PGA	15	8	12	80		6	0	0
GEVO-1	20	12	2	86		7	3	1
Hybrid 1	20	12	2	86		6	1	0
PGA	20	7	12	81		6	0	0

Table 5.2. Comparison of GEVO-1, hybrid 1 and standard small population GA with a population of ten and a generation gap of 0.7.

5.2.1.2 Hybrid Two – GEVO-2

The second dynamic hybrid (GEVO-2) also changed from pure GA to GA with EVOP as the generations increased. Compared to GEVO-1, EVOP was introduced earlier after generation ten, rather than after generation fifteen. Although individual trials were different, the aggregated results which are in classified ranges rather than specific numbers, shown below in table 5.3, show no difference from those achieved using the first dynamic hybrid, GEVO-1, shown in table 5.2.

	No. of GAs compared to initial population						
Generation	**Worse**	**No change**	**Better**		**30%better**	**50%better**	**70%better**
5	7	14	79		2	0	0
10	5	9	86		5	2	1
15	5	9	86		7	3	1
20	12	2	86		7	3	1

Table 5.3. Performance of GEVO2 hybrid GA with a population of ten and a generation gap of 0.7.

The results indicated that the averaged similarities are due to the small population size and few number of generations in these hybrid GAs. A further brief study with much larger populations and the EVOP being introduced to generations much further apart, did show a difference, but the difference between these small hybrids was not sufficient to detect any improvement.

5.2.1.3 Hybrid Three – GEVO-3

The first two dynamic hybrids were used to investigate the effect of introducing EVOP to a GA search during the last few generations. Although counterintuitive, the third dynamic hybrid (GEVO-3) is a valid test hybrid as it is designed to show if changing the search method from GA to combined GA with EVOP, then back to GA, influences the solutions. Results obtained using this hybrid are shown in table 5.4.

	No. of GAs compared to initial population						
Generation	**Worse**	**No change**	**Better**		**30%better**	**50%better**	**70%better**
5	7	14	79		2	0	0
10	12	2	86		5	2	1
15	11	2	87		7	3	1
20	11	2	87		7	3	1

Table 5.4. Performance of GEVO3 hybrid GA with a population of ten and a generation gap of 0.7.

As expected the third hybrid gave a similar performance to the first two dynamic hybrids in the first five generations as all three were purely GA at this stage. For generations ten and fifteen GEVO-3 found more solutions that were worse than the initial population than the other dynamic hybrids, but only a couple of the strings remained unchanged. GEVO-3 found marginally more improved solutions than any of the other GAs, hybrid or standard small population. This result is most probably due to EVOP being introduced earlier in the search than the previous dynamic GAs. It was therefore decided that further investigation into early introduction of EVOP was required before a more general statement can be made.

5.2.1.4 Hybrids Four and Five – GEVO-4 and GEVO-5

These two hybrids will be discussed together as both were used to investigate the effect of using EVOP in increasing numbers of generations as the GA proceeds. With GEVO-4 the EVOP element is quickly introduced in the second generation for one generation, then four generations later for two generations, then four generations later for three generations and finally four generations later for four generations. For GEVO-5 the EVOP element is not introduced until the fifth generation for one generation, then activated for the tenth and eleventh generations, fourteenth, fifteenth and sixteenth generations and finally in the eighteenth, nineteenth and twentieth generations. The summarised results from running these dynamic hybrid GAs at a generation gap of 0.7 is shown in table 5.5.

Generation	No. of GAs compared to initial population						
	Worse	No change	Better		30%better	50%better	70%better
5	15	3	82		2	0	0
10	10	2	88		4	1	0
15	10	2	88		6	2	0
20	10	2	88		6	2	0

Table 5.5. Performance of GEVO-4 hybrid GA with a population of ten and a generation gap of 0.7.

Both dynamic GAs gave similar results, which when averaged and collated into ranges produced almost identical results. As with the other hybrids tested in this section, both GEVO-4 and GEVO-5 were more mobile than the standard small population GA. This mobility resulted in more chromosomes giving worse solutions than the initial population than for any of the other dynamic hybrid GAs. Also GEVO-4 and GEVO-5 produced the least number of chromosomes with no change compared to any of the other dynamic hybrid GAs. The number of chromosomes with no change from the initial population was comparable to those hybrids with EVOP introduced from the first generation. This style of hybrid GA with EVOP only present in selected generations gave the greatest number of improved solutions in all generations although it did not give solutions with the greatest improvement.

5.2.1.5 Hybrids Six and Seven – GEVO-6 and GEVO-7

As hybrids had been designed to test the effect of introducing EVOP to later generations and selected generations throughout, it was considered appropriate to also test the effect of introducing EVOP only to the early generations of the hybrid GA. This would give a clearer understanding of the influence of the local EVOP element on the overall search. GEVO-6 is a GA that uses EVOP in generations one to five only and for GEVO-7 EVOP is present only for generations one to ten. Results of the testing of GEVO-6 and GEVO-7 are shown below in table 5.6 and 5.7 respectively.

	No. of GAs compared to initial population					
Generation	**Worse**	**No change**	**Better**	**30%better**	**50%better**	**70%better**
5	20	2	78	2	0	0
10	17	1	82	5	1	1
15	17	1	82	5	1	1
20	17	1	82	6	1	1

Table 5.6. Performance of GEVO-6 hybrid GA with a population of ten and a generation gap of 0.7.

	No. of GAs compared to initial population					
Generation	**Worse**	**No change**	**Better**	**30%better**	**50%better**	**70%better**
5	23	9	68	2	0	0
10	18	5	77	5	0	0
15	16	5	79	5	0	0
20	16	5	79	5	0	0

Table 5.7. Performance of GEVO-7 hybrid GA with a population of ten and a generation gap of 0.7.

Although for the number of solutions found that were worse than the original population, with both GEVO-6 and GEVO-7, the figures indicate a poor performance compared to the other dynamic hybrids in this section. It should be noted that the number of solutions is similar to those recorded for the original static hybrid (Hybrid 1). There is no significant difference between these hybrids and other GAs in the number of solutions found that appeared in the original population.

Neither of these dynamic hybrids, GEVO-6 and GEVO-7 performed well in finding improved solutions. GEVO-6 with EVOP used only to generation five, gave better results than GEVO-7 with EVOP in generations one to ten. Despite the better performance, the results from GEVO-6 were only marginally better than those obtained with a standard small population GA. These results confirm that combining EVOP with a GA purely in the initial generations is detrimental to the search.

5.2.2 Testing on De Jong's Second Function

As with previous hybrids, these dynamic hybrids were also tested on further test functions including De Jong's second test function. For reference, the results from the comparable standard small population GA (PGA) are shown in table 5.8 and the static hybrid (Hybrid 1) in table 5.9. The dynamic hybrids tested are those listed earlier in table 5.1. Table 5.10 shows the results of testing GEVO-1 on De Jong's second test function.

	No. of GAs compared to initial population					
Generation	**Worse**	**No change**	**Better**	**30%better**	**50%better**	**70%better**
5	9	14	77	49	39	30
10	9	11	80	56	49	41
15	9	11	80	55	49	42
20	8	11	81	55	49	42

Table 5.8. Performance of PGA with a population of ten and a generation gap of 0.8.

	No. of GAs compared to initial population					
Generation	Worse	No change	Better	30%better	50%better	70%better
5	27	3	70	42	35	29
10	24	2	74	50	39	33
15	24	2	74	51	40	34
20	24	2	74	51	40	34

Table 5.9. Performance of Hybrid 1 with a population of ten and a generation gap of 0.8.

	No. of GAs compared to initial population					
Generation	Worse	No change	Better	30%better	50%better	70%better
5	12	11	77	54	46	37
10	9	8	83	60	48	37
15	9	8	83	60	48	37
20	17	0	83	60	52	36

Table 5.10. Performance of GEVO-1 hybrid GA with a population of ten and a generation gap of 0.8

As with previous tests GEVO-1 found more solutions that were worse than the original population but the chromosomes that showed no change were fewer than the standard small population GA. GEVO-1 found more improved solutions than either Hybrid 1 or PGA. The solutions it found were of a better quality than Hybrid 1 but not quite as good as the standard small population GA, as demonstrated by the percentage improvements shown on the right-hand side of the tables.

The results for GEVO-2 are shown in table 5.11. As expected the results for generations five and ten are exactly the same as those for GEVO-1, as the hybrid GAs are identical to this point, however the results then diverge. GEVO-2 finds more solutions that are worse than the original population and has fewer chromosomes with no change. This is especially apparent in generation 15: EVOP is introduced in generation 16 for GEVO-1 and generation 11 for GEVO-2. GEVO-2 finds one less improved solutions than GEVO-1 but they are of a comparable quality.

	No. of GAs compared to initial population						
Generation	**Worse**	**No change**	**Better**		**30%better**	**50%better**	**70%better**
5	12	11	77		54	46	37
10	9	8	83		60	48	37
15	18	0	82		59	51	36
20	18	0	82		59	51	36

Table 5.11. Performance of GEVO-2 hybrid GA with a population of ten and a generation gap of 0.8

GEVO-3 again has a mobile population, but although it gives fewer improved solutions than other GAs, the improved solutions are of a superior quality when compared to the static hybrid or standard small population GA. For the particular set of runs illustrated in table 5.12, movement in the population is almost complete by generation ten.

	No. of GAs compared to initial population						
Generation	**Worse**	**No change**	**Better**		**30%better**	**50%better**	**70%better**
5	11	18	71		56	43	35
10	14	5	81		62	54	45
15	14	5	81		62	54	45
20	14	5	81		62	54	46

Table 5.12. Performance of GEVO-3 hybrid GA with a population of ten and a generation gap of 0.8

Over the large set of trials for the hybrids which combine increasing EVOP in the GA as the generations increase (GEVO-4 and GEVO-5) individual runs differed but gave identical results when averaged and collated into ranges as shown in tables 5.13 and 5.14. As discussed in section 5.2.1.4. this is due to the small number of individuals in a population and the relatively few generations.

	No. of GAs compared to initial population						
Generation	**Worse**	**No change**	**Better**		**30%better**	**50%better**	**70%better**
5	19	0	81		53	49	36
10	14	0	86		62	53	37
15	14	0	86		63	53	37
20	14	0	86		63	53	37

Table 5.13. Performance of GEVO-4 hybrid GA with a population of ten and a generation gap of 0.8.

	No. of GAs compared to initial population						
Generation	**Worse**	**No change**	**Better**		**30%better**	**50%better**	**70%better**
5	19	0	81		53	49	36
10	14	0	86		62	53	37
15	14	0	86		63	53	37
20	14	0	86		63	53	37

Table 5.14. Performance of GEVO-5 hybrid GA with a population of ten and a generation gap of 0.8.

Despite the averaging of results masking some of the features of the hybrids, particularly GEVO-4 and GEVO-5, it was considered important that a large number of runs were necessary for each trial due to the probablistic nature of GAs, as discussed in section 5.1.1. But to analyse this data set of four thousand individual numbers some form of aggregation was necessary. Also it was vital that each trial was arranged and analysed in an identical manner to allow fair comparisons.

The results show that both of these dynamic hybrids have mobile populations, with both GEVO-4 and GEVO-5 registering no chromosomes that have not changed from the initial population. The tables show that these hybrids found more improved solutions than either the standard small population GA or Hybrid 1. However these hybrids show slightly fewer of the most improved solutions.

The remaining two dynamic hybrids, GEVO-6 and GEVO-7, were designed to test the effect of only utilising EVOP during the first few generations, as described in table 5.1. GEVO-7 has a more mobile population than GEVO-6, but finds fewer improved solutions, as shown in tables 5.15 and 5.16. GEVO-6 outperformed both the standard small population GA and Hybrid 1 in terms of the quality of the improved solutions found.

	No. of GAs compared to initial population						
Generation	**Worse**	**No change**	**Better**		**30%better**	**50%better**	**70%better**
5	16	4	80		54	42	33
10	14	5	81		62	54	44
15	15	5	80		61	54	44
20	15	5	80		61	54	45

Table 5.15. Performance of GEVO6 hybrid GA with a population of ten and a generation gap of 0.8.

	No. of GAs compared to initial population						
Generation	**Worse**	**No change**	**Better**		**30%better**	**50%better**	**70%better**
5	23	3	74		50	36	25
10	19	3	78		55	42	34
15	19	3	78		56	42	35
20	19	3	78		56	42	35

Table 5.16. Performance of GEVO7 hybrid GA with a population of ten and a generation gap of 0.8.

Generally these dynamic hybrids gave a performance, which if judged solely on number of improved solutions would indicate a relatively poor performance. However, further analysis of the results show that the dynamic hybrids consistently gave a greater number of superior solutions in the early generations and outperformed the comparable static hybrid in nearly all cases.

5.2.3 Testing on Other Functions

To complete this set of tests the dynamic hybrids were tested on the other three functions in De Jong's traditional test suite: Rastrigin's function, Schwefel's function and Griewangk's function. The results for each function are collated into a table with the solutions found at generations five, ten, fifteen and twenty for each of the seven dynamic hybrids, the comparable static hybrid (Hybrid 1) and standard small population GA (PGA). The seven dynamic hybrids are those described in table 5.1 and discussed in the previous two sections, 5.2.1 and 5.2.2. On the following pages are the results in tables 5.17, 5.18 and 5.19 for Rastrigin's function, Schwefel's function and Griewangk's function respectively.

The GAs recorded in table 5.17 were tested on Rastrigin's function. In general all of the GAs performed well with over 80% of chromosomes returning improved values. However, apart from GEVO-6 the dynamic hybrids returned slightly fewer improved solutions, but of those improved solutions the dynamic hybrids gave more solutions with a greater improvement. The higher mobility of the dynamic hybrid populations initially appeared to be a disadvantage when looking at whether the population had deteriorated, remained unchanged or improved, but closer examination revealed that the dynamic hybrids found more of the higher quality solutions.

Schwefel's function was the subject of the test for the GAs in table 5.18. None of the GAs tested gave a particularly good performance, with no GA finding a solution with an improvement of greater than 30%. The dynamic hybrids did not find as many improved solutions as either the standard small population GA or the static hybrid. The improved solutions found by the dynamic hybrids were of a similar quality to the other GAs but many of the population literally disappeared down Schwefel's dips as this function is particularly deceptive to local search algorithms. The dynamic hybrids gave a poor performance but the best solutions found were as good as those found using the other GAs.

The final set of tests for these dynamic hybrids was on Griewangk's function and the results are shown in table 5.19. In contrast to the previous (Schwefel's)

		No. of GAs compared to initial population					
GA	Generation	Worse	No change	Better	30%better	50%better	70%better
PGA	5	3	6	91	12	7	1
PGA	10	1	5	94	21	11	3
PGA	15	1	5	94	21	11	3
PGA	20	1	5	94	21	11	3
Hybrid 1	5	6	1	93	12	7	3
Hybrid 1	10	6	1	93	19	10	4
Hybrid 1	15	6	1	93	19	10	4
Hybrid 1	20	6	1	93	19	10	4
GEVO-1	5	13	12	75	32	15	5
GEVO-1	10	8	9	83	41	20	8
GEVO-1	15	7	9	84	43	21	8
GEVO-1	20	10	4	86	41	19	9
GEVO-2	5	9	9	82	20	8	2
GEVO-2	10	6	5	89	32	12	4
GEVO-2	15	12	1	87	33	13	4
GEVO-2	20	12	2	86	7	3	1
GEVO-3	5	4	10	86	24	14	6
GEVO-3	10	10	3	87	39	19	8
GEVO-3	15	10	3	87	39	20	9
GEVO-3	20	10	3	87	41	20	9
GEVO-4	5	15	6	79	20	6	1
GEVO-4	10	10	6	84	25	7	1
GEVO-4	15	10	6	84	25	7	1
GEVO-4	20	10	6	84	26	7	1
GEVO-5	5	16	5	79	19	6	3
GEVO-5	10	13	5	82	27	13	7
GEVO-5	15	13	5	82	30	14	7
GEVO-5	20	13	5	82	30	14	7
GEVO-6	5	8	2	90	27	10	3
GEVO-6	10	4	2	94	44	19	9
GEVO-6	15	4	2	94	44	20	9
GEVO-6	20	4	2	94	44	20	9
GEVO-7	5	19	3	78	21	11	4
GEVO-7	10	13	4	83	34	17	9
GEVO-7	15	13	4	83	35	18	10
GEVO-7	20	13	4	83	35	18	10

Table 5.17. Performance of GAs on Rastrigin's function with a population of ten and a generation gap of 0.8.

GA	Generation	No. of GAs compared to initial population					
		Worse	No change	Better	30%better	50%better	70%better
PGA	5	41	6	53	0	0	0
PGA	10	46	4	50	0	0	0
PGA	15	46	4	50	0	0	0
PGA	20	46	4	50	0	0	0
Hybrid 1	5	40	2	58	0	0	0
Hybrid 1	10	44	2	54	0	0	0
Hybrid 1	15	44	2	54	0	0	0
Hybrid 1	20	44	2	54	0	0	0
GEVO-1	5	79	10	11	0	0	0
GEVO-1	10	83	8	9	0	0	0
GEVO-1	15	84	8	8	0	0	0
GEVO-1	20	82	3	15	0	0	0
GEVO-2	5	84	5	11	0	0	0
GEVO-2	10	86	6	8	0	0	0
GEVO-2	15	91	2	7	0	0	0
GEVO-2	20	91	2	7	0	0	0
GEVO-3	5	80	8	12	0	0	0
GEVO-3	10	87	2	11	0	0	0
GEVO-3	15	88	1	11	0	0	0
GEVO-3	20	89	1	10	0	0	0
GEVO-4	5	84	1	15	0	0	0
GEVO-4	10	90	0	10	0	0	0
GEVO-4	15	90	0	10	0	0	0
GEVO-4	20	90	0	10	0	0	0
GEVO-5	5	85	2	13	0	0	0
GEVO-5	10	88	2	10	0	0	0
GEVO-5	15	89	2	9	0	0	0
GEVO-5	20	89	2	9	0	0	0
GEVO-6	5	83	4	13	0	0	0
GEVO-6	10	85	3	12	0	0	0
GEVO-6	15	85	3	12	0	0	0
GEVO-6	20	85	3	12	0	0	0
GEVO-7	5	86	2	12	0	0	0
GEVO-7	10	89	3	8	0	0	0
GEVO-7	15	89	3	8	0	0	0
GEVO-7	20	89	3	8	0	0	0

Table 5.18. Performance of GAs on Schwefel's function with a population of ten and a generation gap of 0.8.

		No. of GAs compared to initial population					
GA	Generation	Worse	No change	Better	30%better	50%better	70%better
PGA	5	3	7	90	20	14	9
PGA	10	2	5	93	26	17	13
PGA	15	2	5	93	26	18	14
PGA	20	2	5	93	26	18	14
Hybrid 1	5	5	5	90	16	13	9
Hybrid 1	10	3	4	93	25	17	11
Hybrid 1	15	3	4	93	25	18	11
Hybrid 1	20	3	4	93	25	18	11
GEVO-1	5	10	15	75	42	24	14
GEVO-1	10	8	11	81	47	35	20
GEVO-1	15	8	11	81	47	35	20
GEVO-1	20	20	2	78	46	33	18
GEVO-2	5	6	11	83	38	21	14
GEVO-2	10	3	8	89	48	34	18
GEVO-2	15	12	4	84	47	31	20
GEVO-2	20	12	4	84	47	31	20
GEVO-3	5	11	9	80	45	28	15
GEVO-3	10	10	2	88	56	39	19
GEVO-3	15	10	2	88	56	39	19
GEVO-3	20	10	2	88	56	39	19
GEVO-4	5	14	7	79	32	19	7
GEVO-4	10	10	6	84	44	28	16
GEVO-4	15	10	6	84	45	30	17
GEVO-4	20	10	6	84	45	30	17
GEVO-5	5	16	4	80	39	22	14
GEVO-5	10	13	5	82	43	27	21
GEVO-5	15	13	5	82	43	27	21
GEVO-5	20	13	5	82	43	27	21
GEVO-6	5	17	2	81	35	21	13
GEVO-6	10	16	2	82	44	31	22
GEVO-6	15	16	2	82	44	31	22
GEVO-6	20	16	2	82	44	31	22
GEVO-7	5	15	2	83	46	30	19
GEVO-7	10	12	2	86	51	42	25
GEVO-7	15	12	2	86	51	42	25
GEVO-7	20	12	2	86	51	42	25

Table 5.19. Performance of GAs on Griewangk's function with a population of ten and a generation gap of 0.8.

function, all GAs gave good results, this is despite Griewangk's function also containing many deceptive local minima. The populations of the dynamic hybrids again gave the most solutions which were worse than the initial population but generally found improvements for more than 80% of the solutions. The standard small population GA and the static hybrid both found the highest number of improved solutions, yet all the dynamic hybrids gave a higher number of solutions with greater improvement. This indicates a higher quality to the population of solutions found by the dynamic hybrids.

5.3 DYNAMIC HYBRID INFLUENCES

5.3.1 Generation Gap

As with the static hybrids in chapter four, the dynamic hybrids were tested over several generation gaps to determine if this setting could affect the solutions found. After previous testing and discussion in section 4.3.2.2 the trials for the dynamic hybrids were conducted over a generation gap of 0.6 to 1.0.

Table 5.20 shows the results obtained from trials using GEVO-1 on De Jong's first function with a population of ten individuals. Mathematical analysis showed that there is no clear correlation between generation gap and improvement in solutions found, although there is a distinct peak at a generation gap of 0.8. This conclusion is in contrast to the findings for the standard small populations discussed in section 4.3.2.2 where there was a relationship between generation gap and quality of solutions found. The changing nature of the EVOP element in these dynamic hybrids is shown to influence performance more than generation gap, which is demonstrated by these results.

		No. of GAs compared to initial population					
Gap	**Generation**	**Worse**	**No change**	**Better**	**30%better**	**50%better**	**70%better**
0.6	5	12	14	74	3	0	0
0.6	10	11	5	84	6	0	0
0.6	15	9	5	86	7	1	0
0.6	20	10	1	89	7	1	0
0.7	5	7	14	79	2	0	0
0.7	10	5	9	86	5	2	1
0.7	15	5	9	86	7	3	1
0.7	20	12	2	86	7	3	1
0.8	5	8	8	84	8	0	0
0.8	10	6	5	89	13	1	1
0.8	15	5	5	90	13	1	1
0.8	20	7	1	92	13	2	1
0.9	5	5	18	77	2	0	0
0.9	10	5	14	81	4	0	0
0.9	15	5	14	81	5	1	0
0.9	20	16	2	82	5	1	0
1.0	5	4	14	82	7	1	1
1.0	10	4	14	82	9	2	1
1.0	15	4	14	82	9	2	1
1.0	20	11	5	84	9	2	1

Table 5.20. Performance of GAs on De Jong's first function with a population of ten.

Since De Jong's first function has a smooth surface area and is the simplest function in the test suite, the lack of correlation found would indicate that no correlation would be found on the more complex surfaces of the other test functions.

An extensive set of tests were not carried out on the other test functions as a sample of trials indicated that there was indeed no correlation.

5.3.2 Effect of EVOP on Search Movement

Several hybrids were created to determine any effect the inclusion of EVOP at various stages of the GA may have on the solutions found. As expected the dynamic hybrids had more mobile populations than the standard small population GAs, but generally the populations were less mobile than those of the static hybrids which included EVOP in all generations. This shows clearly that the EVOP element is affecting the performance of the hybrid GAs and the tests show that the presence of EVOP in a GA gives it more mobility.

The two hybrids with EVOP active only the later generations, GEVO-1 and GEVO-2, gave similar results. Both of these hybrids generally gave more improved solutions than either the standard small population GA or the static hybrid and those improved solutions were typically of a higher quality.

EVOP was only activated for the middle generations of GEVO-3 and this influence could be clearly seen on De Jong's first test function. The population was more mobile in generation ten and fifteen than the other generations with no EVOP present which resulted in marginally more improved solutions being found.

The other dynamic hybrids generally gave good performances with mobile populations which may not find as many improved solutions, but those found were generally of equal or better quality.

5.3.3 Comparison of Dynamic Hybrids

To compare the performance of the dynamic hybrids, each was ranked for its performance on the test functions which is shown in table 5.21. The ranking of the hybrids were then combined to give an overall rank which is shown in table 5.22.

As can be seen in table 5.22 the dynamic hybrids GEVO-3 and GEVO-4 gave the best performances. GEVO-3 contains EVOP from generation six to fifteen and GEVO-4 contains EVOP in generations two, six, seven, eleven, twelve, thirteen, seventeen, eighteen, nineteen and twenty, so both contain a local search element in ten of the twenty generations.

			Function		
Rank	**De Jong 1**	**De Jong 2**	**Rastrigin**	**Schwefel**	**Griewangk**
1	GEVO-4/5	GEVO-4/5	GEVO-6	GEVO-1	GEVO-3
2	-	-	GEVO-3	GEVO-6	GEVO-7
3	GEVO-3	GEVO-1	GEVO-1	GEVO-3/4	GEVO-2
4	GEVO-2	GEVO-2	GEVO-2	-	GEVO-4
5	GEVO-1	GEVO-3	GEVO-4	GEVO-5	GEVO-6
6	GEVO-6	GEVO-6	GEVO-7	GEVO-7	GEVO-5
7	GEVO7	GEVO7	GEVO-5	GEVO-2	GEVO-1

Table 5.21. Ranked performance of dynamic hybrids.

Rank	Hybrid
1	GEVO-3/4
2	-
3	GEVO-1
4	GEVO-5
5	GEVO-6
6	GEVO-2
7	GEVO-7

Table 5.22. Rank of overall performance by dynamic hybrids.

GEVO-2 also contains EVOP in ten of the twenty generations but ranks much lower in sixth place. The main difference between this hybrid and the best performers is that the EVOP is present in the last ten generations. This implies that introducing EVOP earlier into the GA givers better results, whereas it is often assumed that a search should change from global GA to local (Goldberg, 1988(b), Renders and Bersini, 1994).

The relatively poor performance of GEVO-6 and GEVO-7 indicate that EVOP is of little value if only used in the early stages of the search. GEVO-2 is perhaps one of the more surprising results as the similar hybrid GEVO-1 ranks third whereas GEVO-2 ranks sixth. This indicates that EVOP does have a value as the search progresses, but as earlier discussion concludes, early introduction of EVOP with some EVOP in the later stages is the best combination. EVOP does not need to be present in every generation of the GA but its presence enhances the quality of the solutions found.

5.4 SUMMARY

Experimentation has shown that the generation gap has little influence on these dynamic hybrids. As discussed earlier in Chapter 4, the generation gap was shown to have an influence on the simpler functions for the standard small population GA, but this influence was much less apparent with the static hybrids. This was attributed to the presence of EVOP. Since EVOP is also present in these dynamic hybrids but in a less stable manner, the influence of the generation gap has been overpowered by the influence of EVOP.

By comparison of the dynamic hybrids to both the standard small population GAs and the static hybrids, it was demonstrated that the EVOP local search element influences the mobility of the search. EVOP was shown to make the search more active, ensuring even with small populations and few generations that there is greater change from one generation to the next. This is why the static hybrids where more mobile than the dynamic hybrids which contained less EVOP.

The performance of several different dynamic hybrids were compared over the five test functions used throughout these studies to determine the optimum combination of local and global, EVOP and GA, search. The results demonstrated that using the EVOP local search only in the early generations was not of value. The best performing hybrids had EVOP present in half of their generations, but rather than running the global GA search then introducing the local EVOP search, the best combination included an early introduction of EVOP with some EVOP also present in the later generations. EVOP does not need to be present in all generations of the search, but its presence enhances the quality of solutions found.

The limited availability of published works for comparison of the small population hybrids tested in both this and the previous chapters, coupled with the lack of detailed breakdown of conclusions drawn from many experiments, for direct evaluation led to the consideration of the value of benchmarking for GAs, which is explored in the next chapter.

6. BENCHMARK TESTING AND APPLICATIONS

The aim of this chapter is to discuss the benefits and problems of benchmarking and its applicability to GA and GA hybrid research. The current use of benchmarking for GAs is then examined, before an illustration of the investigations undertaken to examine the performance of the hybrid GAs on a range of further 'benchmark' test functions. The areas of application for small population GAs are discussed before potential applications for the hybrids tested in this study are considered and implementation illustrated by an example.

6.1 BENCHMARKING

Benchmarking has recently become a prominent issue for the manufacturing sector, although the practice of benchmarking has been conducted for many years in some form in most sectors of industry. The increased acceptance, or in some cases insistence for companies to perform to International Standards, such as the ISO14000 series or QS9000, has raised awareness of the need to measure performance. Standards often state actual values or specific dimensions that must be achieved for a product to be certified as reaching the required standard. External auditing ensures that the records and methods of measurement meet the required regulations.

However many standards, more especially those published recently, e.g. ISO9000 series, do not state a specific dimension or value to be attained but specify a process for monitoring, maintaining and auditing records. The move to more comprehensive performance measurement by industry, fuelled by the 'quality movement', has led to a wider interest in benchmarking, as measured performance does not indicate the relative position of a company or its product(s) in the market place unless a comparison is made to competitors. Unfortunately as industrial competitors are obviously in competition with each other, there is a great unwillingness to share information on performance. This is one of the great hurdles facing benchmarking in many areas. Another area of concern is that all data must be comparable, accurate, reliable and auditable, since it is highly unlikely that a single person has collected or been able to fully audit all the data. Another issue is the fact that no two manufacturing units are exactly the same, prohibiting absolute direct comparisons.

6.1.1 Benchmarking for GAs

These problems also face the GA research community. Although academia has a greater tendency to share information than industry there is no widely accepted standard benchmark for GAs. As described earlier in Chapter 4, the best known 'benchmark' for GAs is De Jong's test suite of five functions (De Jong, 1975) which have been widely used and appear in some of the works which provide the foundations to many people's interest in GAs, such as Goldberg (1988b). The functions provided an early base for GA performance comparison, but as GA research has matured and the applications have become more complex and diverse, these functions are no longer as prominent as they once were. Although understanding of GAs has progressed there is no definitive method of producing a competent GA. By not comparing a large amount of work, i.e. benchmarking, the GA community may be neglecting an area which could lead to a better understanding of how GA parameters influence performance.

A GA benchmark would require a range of functions at a set number of dimensions. In addition to these standard functions, it would be useful to study a range of application domain specific functions where applicable. The travelling salesman problem (TSP) is perhaps one of the best known examples of a GA benchmark test. Most fields of engineering have standard problems which students and practitioners regard as fundamental or contain the essence of an area of study. For example, scheduling has standard problems (Zalzala and Fleming, 1997, Muth and Thompson, 1963) such as the 3 *x* 3, 10 *x* 10, 20 *x* 20, M *x* N etc. where M is the number of machines and N is the number of jobs.

A GA benchmark would require more than a set of standard functions for testing, as there are many other influences on GA performance. As well as the quality of an answer, the time taken to reach a solution is also important to industry. The importance of the speed of a GAs varies with the application, but a wider range of uses would be possible if GAs were generally faster in implementation. Little published work gives information regarding the time taken to find a good or optimal solution or to complete a certain number of generations.

GA run time is also heavily influenced by the computer processor, platform and software used. It would be impractical for a benchmark to insist on a particular type of computer processor to be used at a certain speed or operating system, as the standard would soon be obsolete due to the rate of progress in computing science. Indeed the work for this study has been carried out on four different desktop PCs, two laptops and four different operating systems. Similarly, insistence on using particular software would restrict innovation. A standard method of reporting, which includes time taken, type of processor, software and the minimum computing requirements as well as the important issue of algorithm performance, such as quality of results found, would allow researchers to begin to compare their GAs.

The above details coupled with more widely published information on GA parameters such as population size, crossover operator or selection operator used could lead to a deeper understanding of GAs. There would be a wider industrial acceptance to use GAs if there is an audited body of evidence of the control parameters of GAs, as industrial investment money is often tightly wrapped in caution. The general public tend to believe that computers and software should be logical, repeatable and find 'the' answer every time and anything which does not, cannot therefore be reliable. One of the greatest hurdles for GAs is the psychological problem of being able to accept that the computer will find a good solution in a manner that is not exactly repeatable and that the computer may not find 'the' best answer every time.

In an ideal world there would be anonymous comparative studies to determine influences on GAs collected in a database, but this is probably infeasible due to many factors including funding issues. The onus is therefore on individual researchers or research groups to publish detailed findings to help with an understanding of GAs that could reach further than their own specific areas of research.

6.1.2 Benchmarking for EVOP

EVOP is an experimental design method that prescribes a method of calculating an answer. The only area where differences may occur is in the step size. Companies wishing to ensure that all plants use the same parameters could issue guidelines, such as Ford, General Motors and Chrysler who set out how to conduct an EVOP experiment in a QS-9000 handbook. As there are no other factors that can be changed to influence an EVOP experiment there is not a great need to benchmark the process. This is not to say that individual areas of application of the method cannot be benchmarked but that the methodology itself does not require a benchmark.

6.2 TESTING OF GAs

The hybrid GAs in this thesis were tested on a range of functions for which some standard data had been collated. Lazauskas (1999) lists several well known functions such as Rastrigin's and the weighted sphere functions along with the mean number of function evaluations required to attain 'roughly three digit accuracy'. Although only limited data is available from Lazauskas (1999) it is one of the few information sources that states GA results rather than benchmarking the actual test functions. Unfortunately the listed results were for large population algorithms, so it is not possible to make a direct comparison to the small population hybrid GAs studied in this thesis. However, since the functions are benchmarked it is valid to test the hybrid GAs created during this study to ascertain their performance.

6.2.1 Selection of Hybrid GAs for Testing

Three GA hybrids from the previous two chapters were selected for testing on this extended range of functions. The first hybrid is a static hybrid, referred to as EVOS, with a generation gap of 0.7, a population of twenty individuals and running to twenty generations. The other two hybrids are dynamic hybrids referred to as GEVO-3 and GEVO-4 in chapter five, as these two hybrids gave the best performances of all the dynamic hybrids tested over the range of functions. Both of these GAs are also tested with population sizes of twenty individuals and run to twenty generations. EVOP was added to these dynamic hybrids in the manner specified in section 5.2, which related to hybrids with no more than twenty generations. As the hybrids were to be tested with many more generations, these schemes were extended. GEVO-3, which has EVOP after the first and before the last five generations in a twenty generation hybrid, was redefined as having EVOP in the middle 50% of generations. GEVO-4, which has a scheme of more EVOP gradually being invoked, was extended in the same pattern and after the one hundredth generation every generation would contain EVOP. To complete the set of GAs a standard small population GA was also tested. This GA has a population of twenty with a generation gap of 1.0, referred to as SPGA.

6.2.2 Benchmark Functions

Four functions were selected from Lazauskas (1999), as the other functions were incompletely described. The benchmark functions were:

1. The Weighted Sphere Model $f(x) = \sum_{i=1}^{n} ix_i^2$ $-5.12 \leq x_i < 5.12$

at n = 3 and n = 30

global minimum = 0 at $x_i = 0$)

which is 'considered easy for GAs' although at n = 3 the best GA listed was a traditional GA which took 805 evaluations to reach roughly three digit accuracy and for n = 30 a traditional GA reached a solution of approximately 5.0 after 25000 evaluations.

2. Rastrigins' Function $\qquad f(x) = n * A + \sum_{i=1}^{n}(x_i^2 - A * \cos(2\pi x_i))$

   $-5.12 \le x_i \le 5.12$

   A=10.0 $\qquad$ (global minimum = 0 at $x_i = 0$)

which is 'considered as difficult for most methods'. This is one of the functions extensively used in the previous two chapters. Lazauskas (1999) reports that three GAs reached a solution of 0.9 with between 3608 and 9900 evaluations at n=20. Others GAs failed to reach this level of accuracy but a traditional GA reached a solution of 45.0 after 25,000 evaluations.

3. Schwefels' Function $\qquad f(x) = V * n + \sum_{i=1}^{n} - x_i * \sin\left(\sqrt{|x_i|}\right)$

   $-500 \le x_i \le 500$

   V=418.9829 $\qquad$ (global minimum = 0 at x_i = 420.9687)

which is a difficult function as the 'second best' minimum is a long way from the global minimum and some algorithms can become trapped in the wrong region. This function is also one used extensively in the previous two chapters. This function was tested at n=10, with Lazauskas (1999) reporting GAs requiring approximately 100,000 or 200,000 evaluations to reach three digit accuracy.

4. Griewangks Function

$-600 \leq x_i \leq 600$

$$f(x) = 1 + \sum_{i=1}^{n} \frac{x^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right)$$

(global minimum = 0 at $x_i = 0$)

No description is given of this function, but it has many local minima and only four sets of records are listed. A two dimensional plot of this function can be seen in chapter 4, section 4.4.1.

The results reported in Lazauskas (1999) for each function raises the question of the starting points or seeding of the initial populations, which could also be stated in a GA benchmark: A standard set of random numbers, from the standard tables published in many mathematical text books or how seeding was achieved. Seeding is an area of active research, e.g. Lee and Rowlands (1998).

6.2.3 Performance of Hybrid GAs

For the weighted sphere model, the four GAs were tested at the settings shown in table 6.1. n is as specified in section 6.2.2. A summary of the results obtained on this model are given in table 6.2.

Setting	**N**	**Number of generations**
1	3	20
2	3	800
3	30	20
4	30	800

Table 6.1. Settings tested on the weighted sphere model.

Setting	GA	Best solution found: Overall	in last generation	Time for 100 runs (seconds)
1	GEVO-3	2.024101	2.024101	36.14
1	GEVO-4	1.932815	1.932815	59.81
1	EVOS	2.063431	2.063431	58.50
1	SPGA	6.949797	6.949797	61.08
2	GEVO-3	1.618897	1.775366	1606.20
2	GEVO-4	2.163712	2.163712	2650.40
2	EVOS	2.322033	2.322033	2317.20
2	SPGA	2.488694	2.488694	588.31
3	GEVO-3	1.131182	1.407389	36.20
3	GEVO-4	2.245126	2.245126	58.99
3	EVOS	2.526541	2.526541	58.77
3	SPGA	6.938774	6.938774	58.44
4	GEVO-3	2.040263	2.040263	1599.60
4	GEVO-4	1.525893	1.525893	2663.80
4	EVOS	1.574880	1.574880	5723.60
4	SPGA	6.860670	6.860670	3119.50

Table 6.2. Summary of results obtained on weighted sphere model.

The benchmarks available stated for n=3 (settings 1 and 2) that a traditional GA took 805 evaluations to reach three digit accuracy. None of the hybrids managed to find the optimum, but GEVO-4 gave the best performance with only twenty generations and the standard small population GA the worst performance. When the GAs were run to eight hundred generations the performance of GEVO-4 and EVOS deteriorated slightly, although both recorded a better result than the standard small population GA and GEVO-3 gave the best performance.

At n=30 GEVO-3 and GEVO-4 again gave the best performances, with GEVO-3 returning the fastest times. Increasing the number of generations improved the performance of the standard small population GA (SPGA). The performance of the hybrids varied but did not show a great difference for the time spent running the extra generations. The benchmark reached the minimum after 40,000 evaluations, but since the philosophy of the hybrids is simplicity and speed they were not evaluated at this number of generations.

For Rastrigin's function at n=20, as specified in section 6.2.2, GEVO-3 gave the best performance in the quickest time. The results given in table 6.3 reinforce those findings stated in chapter five, where GEVO-3 gave a much better performance than GEVO-4. GEVO-3 gave a performance that was better than those cited in the benchmark in fewer generations, twenty compared to over six thousand, but the other hybrids did not achieve as good solutions. Perhaps the most significant figure is the speed of operation of GEVO-3. GEVO-3 was more than two and half times faster than the other hybrids, and completed one hundred run in less than seventy five percent of the time taken by the standard small population GA (SPGA).

GA	**Best solution found:**		**Time for 100 runs (seconds)**
	Overall	**in last generation**	
GEVO-3	0.2058439	0.205843	28.62
GEVO-4	2.762299	2.762299	75.36
EVOS	2.762300	2.762300	76.19
SPGA	2.047870	2.047870	39.71

Table 6.3. Summary of results obtained on Rastrigin's function.

Schwefel's sine root function was tested at n=10. One benchmark GA quoted 100,000 evaluations to reach the global minimum and the other at 8699 evaluations, although the optimum was not found in four out of fifty runs. The results reflect the difficulty in optimising Schwefel's function with the second best minima distance from the global minimum. This challenge was also reflected in the results reported in section 5.2.3. This function at greater values of n often requires more than a million iterations to reach the global minimum with three digit accuracy. Table 6.4 shows the results obtained by the hybrids on Schwefel's function.

GA	**Best solution found:**		**Time for 100**
	Overall	**in last generation**	**runs (seconds)**
GEVO-3	2.54285	2.92150	37.24
GEVO-4	2.37426	2.97632	58.11
EVOS	2.54285	2.97633	62.72
SPGA	2.67640	3.13214	27.52

Table 6.4. Summary of results obtained on Schwefel's function.

Table 6.5 shows the results obtained on Griewangk's function with n=10. The benchmark GAs are reported to take approximately 100,000 evaluations to reach 0.1 with a standard GA, with a coevolutionary GA "doing better". Another GA (Muhlenbein *et al.*, 1991) is quoted to find the minimum to three digit accuracy in 59,520 evaluations. GEVO-3 and GEVO-4 gave similar performances, with both the standard small population GA and the static hybrid returning better performances in table 5.19 on Griewangk's function. The standard small population GA gives the best performance in the shortest time for twenty generations, although no GA finds the minimum. The results stated in table 6.5 are for the first one hundred runs recorded for each hybrid. Further testing produced slightly different results, but there was no improvement in the best solutions found. The repeating of 2.00432 suggests that the hybrids became trapped at this value.

GA	**Best solution found:**		**Time for 100**
	Overall	**in last generation**	**runs (seconds)**
GEVO-3	2.00432	2.00432	38.44
GEVO-4	2.00432	2.00432	59.98
EVOS	2.00432	2.00432	59.82
SPGA	1.10433	1.10433	25.65

Table 6.5. Summary of results obtained on Greiwangk's function.

6.3 APPLICATIONS OF SMALL POPULATION GAs

Applications of traditionally sized GAs continue to grow in many different fields from circuit design to medicine. Research regarding small population GAs or micro GAs is on a much smaller scale, e.g. Goldberg (1989) and Goldberg et al. (1991) considered the sizing of populations. The relatively small amount of work published regarding the theory of small population GAs is reflected in published work regarding application of this technique. Reeves (1993) discuss the theory of small population GAs and suggests possible applications such as engineering design where the effect of a given number of parameters has to be determined by experiment and current methods, such as design of experiments, make simplifying assumptions. Small population GAs are suggested as a possible replacement for moderately sized design of experiment techniques. Dozier et al. (1994) used small population GAs to solve the N-Queens problem, where the challenge is to place N Queens on a N *x* N chessboard so that they cannot attack each other. Chen and Wu (1998) used relatively small population GAs for channel and data estimation, but the GAs run to thousands of generations.

6.3.1 Applications for Hybrid GAs

Hybrid GAs could potentially be used in areas that standard GAs have proved to be useful and there are many example applications of larger population hybrid GAs reported in recent conferences. The very small hybrid GAs studied here could also potentially be used in these areas and where time constraints currently make traditional GA application impractical.

Industry is yet to be widely convinced of the value of genetic algorithms or evolutionary computation in general (Poli, 1999). The small amount of work currently relating to small population GAs will hinder their adoption in the workplace.

Applications suitable for the hybrid GAs studied in this thesis include small scheduling or re-scheduling problems, e.g. a factory schedule completed by a standard large population GA on a weekly or daily basis, but if say 10% of the workforce phone in sick half an hour before the start of a shift and there is not sufficient time to run the full GA, potentially a hybrid small GA could be used to optimise a small portion of the schedule.

Monitoring a poorly or vaguely understood manufacturing process or machine with varying output is often monitored using design of experiments but this makes assumptions about the interactions of the parameters. Hybrid GAs could be used in this application especially if the parameters are constantly changing. This second suggestion is potentially the easier to code and test, providing a suitable machine and process is found.

6.3.1.1 Example implementation

To demonstrate that it is possible to apply a hybrid GA to an optimisation problem, sample tests were carried out and a small demonstration of implementing a hybrid GA is given below using a version of the travelling salesman problem.

The travelling salesman problem is generally the problem of determining a route for a salesman to visit each of N cities with given positions (x_i, y_i) once, and only once, before returning to the home (start) city. This problem is NP-complete and so the implementation of a hybrid GA is demonstrated with a small number of cities, although it is possible to increase the number of cities for these hybrid GAs.

For this simple demonstration there are four cities, A, B, C and D, with the co-ordinates as shown in figure 6.1.

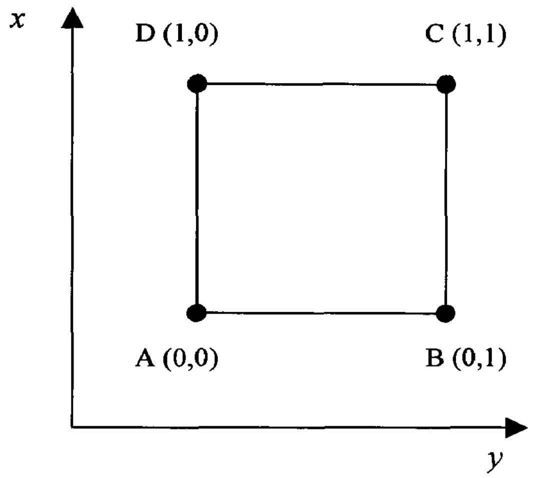


Figure 6.1 Positions of the cities.

Although real value representation is better suited to this particular problem with large values of N (number of cities), it is merely being used as a demonstration of implementation with an extremely low value of N, so binary representation is used, as discussed in Chapter 4. The path is represented in chromosomes of eight bits length as shown in figure 6.2. The first two bits represent the position of city A in the path, the second two bits the position of B in the path etc.

Chromosome	1	1	0	1	0	0	1	0
City	A		B		C		D	
Position in path	third		first		start		second	

Table 6.6 Path representation

Although in the initial population the two 'start' bits could be viewed as redundant, in future generations they will retain the start points and so information will not be lost. With this form of representation several cities could be allocated to the same position in the path. This is overcome by using the left side of the chromosome as dominant. The position of city A in the path will always be as described by the chromosome, if city B is allocated the same position in the path as city A then the position will increase by one, a parallel with the operation of EVOP. If city C is allocated the same position as any of the previous cities then the position is increased by one, if this position is also occupied, then the original position of C is decreased by one, if this position is also occupied then the step size is increased and the method repeated until C is allocated a position in the path. This pattern also reflects the operation of EVOP and is continued until all cities have been allocated a position in the visit path.

The hybrids were capable of finding the optimum path for low numbers of cities (ten or less) but as the number of cities increased the representation becomes more inefficient. The principle of operation works but real valued representation is a more efficient method and these hybrids are not designed or expected to solve large problem spaces. As discussed in section, 6.3 small population GAs are not applicable for problems, such as the travelling salesman problem, that suit large population GAs. The main reason for this test was to demonstrate that the combined method could be applied to other domains than those demonstrated earlier.

6.4 SUMMARY

Currently there is only a small number of benchmarks available and these are for large population GAs. The hybrid GAs tested here showed varying degrees of success, but it is difficult to give an accurate reflection of their performance as they were not "like for like" tested with other small population or hybrid GAs. Benchmarking of GAs has great potential to widen the understanding of the theory of the operators used by GAs provided a standard reporting format is adopted.

It is recommended that a GA benchmark would include a range of standard test functions, such as the De Jong test suite, at set dimensions and where applicable domain specific functions, such as the M *x* N scheduling problems. In addition to the settings used for the GA parameters, including population size, number of generations, crossover method, selection operators etc., a benchmark would also require a report on the computer processor, platform and software used and run times of the algorithms.

Small population GAs command only a small proportion of the research in the GA community and the number of applications reflects this. These GAs have potential some areas of application, but they may become more applicable if interaction of GA operators becomes better understood. The suggested benchmarking process could address this issue.

7. CONCLUSIONS AND FURTHER WORK

7.1 CONCLUSIONS

The review of evolutionary computation and experimental design, in chapter two, identified genetic algorithms (GAs) and evolutionary operation (EVOP) as optimisation techniques that could be combined to develop a hybrid method which incorporates both global and local search ability. Further study of these techniques gave an understanding of their operation and parameters that could influence their performance.

An investigation into the software available to implement GAs and EVOP, described in chapter three, revealed that the newer technique, GAs, had many suitable software programs available either commercially or as development tools. The older technique of EVOP was only available in a couple of experimental design packages and solely as Simplex EVOP. An initial program was written in C++ for two factor EVOP. Analysis of the C++ programming process indicated that EVOP could be implemented on a spreadsheet. Two and three factor EVOP can now be carried out on the new software developed for this study, AutoEVOP, a program running in Microsoft Excel. Writing this program indicated that the algorithm could be encoded onto a relatively small computer chip with appropriate sensors or actuators for an inexpensive on-line automatic process optimiser. Due to the objective of creating a relatively fast optimisation technique with the potential to be applied in an industrial setting, the criteria for selecting the software included the requirement for any software to run on a 'standard' PC. A technique that requires expensive equipment or is time consuming is less likely to be used or developed, as demonstrated with the history of the EVOP method. The GA Toolbox was selected as it was inexpensive software that was flexible for experimentation yet ran in a widely available environment, Matlab, which runs on standard PCs with a Windows operating system.

The next step was to choose appropriate settings for the GA. This was discussed and results of experiments presented in chapters four and five. Binary had been shown to be the best form of coding for small population GAs and was used here as it also enabled a guaranteed step size for the EVOP element of the proposed hybrids. Any resulting algorithm was required to quickly reach a solution, so small population GAs with thirty or less individuals per generation were selected as the base for the new hybrids. The EVOP element was included as it would assist with preventing the small population GAs from prematurely converging. Testing of the new hybrids and conventional small population GAs was on a set of well known and documented functions that covered various types of search space from smooth simple curves to more complex spaces with many local minima.

As little literature exists relating to suitable settings for the very small population GAs studied in this thesis, a range of tests were undertaken which determined that stochastic universal sampling was a more suitable selection method than the traditional roulette wheel tournament. As the initial chromosomes were relatively short, single point crossover was selected. The simplest method of crossover reflected the EVOP philosophy and was quickly executed as code. As the population contained a small number of individuals, mutation operating at normal rates of one in a thousand or greater would have very little effect. Greatly increasing the mutation rate would risk turning the search into a random walk, so this operator was switched off. The EVOP adds an element of controlled mutation, so reducing the risks associated with not using mutation. The hybrid GAs outperformed the standard small population GAs and gave results in a much quicker time than standard large population GAs.

A set of experiments were conducted to determine any influence on the performance of the GAs by the value of the generation gap. Testing was carried out on the complete set of functions, but initially concentrating on the first two functions as they are the simpler spaces which allowed a clearer picture of the behaviour of the GAs. Generation gap, although dependent on search space, influenced the performance of the GAs. Improved solutions were generally found with higher generation gaps. The results of the empirical testing led to the recommendation that a generation gap of 0.7 should be used as a starting point for experimentation with small population GAs and their hybrids. All of the above experiments were also carried out with various population sizes. GAs with a larger population size giving, as expected, the best results, although this naturally increased the required run time. For all types of GA tested a population of five individuals was too small to produce acceptable results. Most further experimentation was carried out with populations of twenty individuals as these GAs were capable of improving the solutions found, yet were small enough to have a quick run time.

As previous testing had demonstrated that the addition of EVOP can improve the solutions found by a small population GA, further testing was carried out in chapter five to determine the influence of EVOP and if its presence was of greater importance in particular generations. Several hybrids with EVOP present in various combinations of the generations were tested. As anticipated these particular hybrids, referred to as dynamic hybrids, were not as mobile as the static hybrids with EVOP in all generations, but more mobile than the standard small population GAs. The hybrid GA with EVOP present in the middle generations and another with EVOP present in progressively more generations gave the best results of the dynamic hybrids tested. The results demonstrated that combing EVOP only in the early generations of a GA is of little value. Convention would suggest that the search should change from global to local as it progresses, but these tests show that the hybrids give better solutions with the early introduction of EVOP and some present in later generations. EVOP does not need to be present in every generation of the GA but its presence enhanced the quality of the solutions found. With the dynamic hybrids the influence of the generation gap was less significant, as it was overpowered by the influence of EVOP.

The experimentation undertaken raised the issue of benchmarking GAs which is discussed in chapter six. Although the hybrid GAs in this thesis had been tested on a range functions commonly found in the literature, publishing of this type of results information is not as common as it once was. This may be due to traditional style GAs now being used for applications, but important theory as to the precise nature of the influence of GA operators remains unknown and may remain so unless comparative testing, benchmarking, is undertaken.

To enable an understanding of the parameters which should be included in any future GA benchmark some of the better hybrids developed earlier in this study were benchmarked on a range of further functions. The study gave an indication of the difficulties in obtaining benchmark information or comparable results from more than one publication. The hybrids were tested on specific functions and timed for one hundred runs. The testing resulted in the following recommendation for a GA benchmark: standard set of test functions, such as De Jong's suite, additional application specific functions where appropriate; processor; platform; software used; as well as the traditional GA operator settings such as population size, number of generations, crossover operator and selection methods. A more detailed and consistent method of reporting would enable a database of cross referenced works to be established to aide a deeper understanding of the influence of GA parameters.

7.2 CONTRIBUTIONS

This thesis investigated the areas of evolutionary computation and experimental design to determine a novel method which combines both global and local search capabilities to find good, but not necessarily the best answer in a short length of time. To meet this objective there were several stages to the research which included the contributions described below.

A new piece of software named AutoEVOP was developed to implement two or three factor EVOP running on Microsoft Excel. Users can manually enter the names of the factors to be studied and the software will indicate the settings required to complete an EVOP experiment. Readings from the experiments are then entered and AutoEVOP will indicate if further experimentation is required or if a stop condition has been reached.

Several new GA - EVOP hybrids, which contain elements of global and local search, were created for this study. Static hybrids combined GA and EVOP searching in all generations, whereas the dynamic hybrids contained EVOP in a controlled number of generations.

For very small populations, twenty or less individuals, stochastic universal sampling was demonstrated to be the most suitable method of selection, rather than the more traditional roulette wheel selection method. The performance of very small population EVOP hybrid GAs was shown to improve with larger generation gaps on simple functions and on more complex functions increasing the generation gap does not deteriorate performance. As a result of the testing carried out for this study a generation gap of 0.7 was recommended as a starting point for empirical searches using small population GAs and their hybrids. Due to the changing presence of EVOP, the generation gap has less influence on dynamic EVOP – GA hybrids compared to the static hybrids.

The EVOP local search element was shown to positively influence the performance of the small population GA search. The EVOP operator in the hybrid GA gave the greatest improvement in performance when present in the middle generations or with a progressively greater presence.

A recommendation of the information required to be reported for benchmarking GA performance is also presented. This includes processor, platform, software information as well as GA parameters such as population size, number of generations, crossover method and selection operators and results of testing on a set of standard test functions.

7.3 FUTURE WORK

The range of applications for standard large population GAs continues to grow, but the acceptance of GAs in industry as an optimiser will be hindered until there is a greater understanding of the elements that control GA performance. Small population GAs are a niche area of GAs and as yet have limited application, but as understanding of GAs expands smaller populations have the potential to become the norm as large time consuming populations testing vast areas of the search space become unnecessary.

The current lack of use of the original EVOP technique could be due to the technique requiring a large number of repetitive calculations which prevented its use before the advent of widely available computing power in the form of the desktop PC. EVOP continues to have a small presence in many experimental design books, but despite being included in QS-9000 it continues to be under utilised. This may remain the case unless large manufacturers insist rather than suggest suppliers use EVOP or EVOP is coupled with modern shop floor data collection techniques and possibly integrated with expert systems and / or fuzzy logic rule bases.

The work carried out for this study should be extended by further testing of the hybrid GAs to establish the most efficient crossover operator. The hybrids would benefit from further evaluation on actual applications, although small population GAs currently have a very restricted repertoire, as discussed earlier. An evaluation of the criteria used to determine the suitability of a problem for GAs would be useful. Matlab is a useful tool and found in universities across the world but it is not as common in industry. The code for the hybrids could also be written in another language such as C or C++ which is commonly found in industry to allow experimentation on real industrial data without the need for re-entering large amounts of data. Although this study concentrated on small population GAs it would be interesting to apply the EVOP hybridisation technique to other applications using large population GAs.

8. REFERENCES

ALBA, E.; ALDANA, J. F., and TROYA, J. M. 1993. Genetic Algorithms as Heuristics for Optimizing ANN Design. *Artificial Neural Networks and Genetic Algorithms, Proceedings of the International Conference. Innsbruck, Austria; 1993.* Wein, Austria: Springer Verlag. pp683-690.

ALVAREZ, L.F. 2000. Application of Genetic Programming to the Choice of a Structure of Multipoint Approximations. http://www.student.brad.ac.uk/lfalvere/ papers/issmo/issmo.htm (accessed December 2000)

anonymous. 1961. Evolutionary Operation in Plant-Scale Experiments. An attitude or a technique? *Industrial and Engineering Chemistry*, **53**, pp36A-41A.

anonymous. 1999. What is Genetic Programming? http://www.genetic-programming.com/gpanimatedtutorial.html (updated October 1999)

ARABAS, J.; MICHALEWICZ, Z., and MULAWKA, J. 1994. GaVaPS - a Genetic Algorithm with varying Population Size. *Proceedings of the First IEEE Conference on Evolutionary Computation: IEEE World Congress on Computational Intelligence,* 27th June 1994. Orlando, Florida. New Jersey, Piscataway. pp 73-78.

BÄCK, T. and SCHWEFEL, H-P. 1993. An Overview of Evolutionary Algorithms for Parameter Optimization. *Evolutionary Computation.* **1**(1), pp 1-23.

BARNETT, E. H. 1960. Introduction to Evolutionary Operation: a method for increasing Industrial Productivity. *Industrial and Chemical Engineering.* **52**(6), pp500-503.

BISGAARD, S. 1994. Blocking generators for small 2^k-p designs. *Journal of Quality Technology.* **26**(4), pp288-296.

BISHOP, J. M., BUSHNELL, M. J., USHER, A., and WESTLAND, S. 1993. Genetic Optimisation of Neural Network Architectures for Colour Recipe Prediction. *Artificial Neural Networks and Genetic Algorithms, Proceedings of the International Conference,* Innsbruck, Austria. Wein, Austria: Springer Verlag. pp719-725.

BONOMI, E., LUTTON, J-L. 1984. The N-city traveling salesman problem: statistical mechanics and the Metropolis Algorithm. *SIAM Review.* **26**(4), pp551-569.

BOX, G. E. P. 1957. Evolutionary Operation: a method for increasing Industrial Productivity. *Applied Statistics, The Journal of the Royal Statistical Society, Series C.* **VI**(2), pp81-101.

BOX, G. E. P. 1988. Signal to Noise Ratios, Performance Criteria, and Transformations. *Technometrics.* **30**(1), pp1-40.

BOX, G. E. P., BISGAARD, S. and FUNG, C. 1988. An Explanation and Critique of Taguchi's contribution to Quality Engineering. *Quality and Reliability Engineering International*, **4**(2), pp121-131.

BOX, G. E. P. and DRAPER, N. R. 1969. *Evolutionary Operation: a Statistical Methodfor Process Improvement.* New York, London: J.Wiley and Sons.

BOX, G. E. P., HUNTER, W.G. and HUNTER, J.S. 1978. *Statistics for experimenters: an introduction to design, data analysis and model building.* New York, USA: J. Wiley & Sons. pp362-373.

BREYFOGLE, Forrest, W. III. 1999. *Smarter Solutions Using Statistical Methods.* New York, USA: J. Wiley & Sons.

CAPPONETTO, R., FORTUNA, L., GRAZIANI, S., and XIBILIA, M. G. 1993. Genetic algorithms and applications in system engineering: a survey. *Transactions of the Institute of Measurement and Control*, **15**(3), pp143-156.

CARLEYSMITH, S.W. 1994. Industrial Fermentation Control: What's driving progress? *Proceedings of the Second Conference on Advances in Boichemical Engineering.* Institution of Chemical Engineers, pp 45-7. ISBN: 085295333X.

CHATTO, K. A. and KENNARD, R. W. 1961. Evolutionary Operation in Plant-Scale Experimerts - the Simplified Concepts. *Industrial and Engineering Chemistry*,

53, pp42A-45A.

CHEN, H.P. 1989. A Practical Approach for using Design of Experiment in Die Casting. *North American Die Casting Association 15th International Die Casting Congress and Exposition*; 16th October 1989, St Louis, MO., USA. Paper no. G-T89-123.

CHEN, S., and WU, Y. 1998. Maximum Likelihood Joint Channel and Data Estimation Using Genetic Algorithms. *IEEE Transactions on Signal Processing.* **46** (5). pp 1469 – 1473.

COLEMAN, D. E. and MONTGOMERY, D. C. 1993. Systematic Approach to Planning for a Designed Industrial Experiment. *Technometrics,* **35**(1), pp1-12.

CORNELL, J. A. 1990. *Experiments with Mixtures: Designs, Models, and the Analysis of Mixture Data.* 2nd edn. New York: J.Wiley and Sons.

DAVIS, L. 1987. *Genetic Algorithms and Simulated Annealing.* London: Pitman. 1987.

DAVIS, L. 1991. Bit-climbing, Representational Bias and Test Suite Design. *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp18-23.

DE JONG, K. 1975. Analysis of the behaviour of a class of genetic adaptive systems, PhD Thesis, University of Michigan.

DE JONG, K. 1993. Editorial Introduction. *Evolutionary Computation.* **1**(1), pp iii-v.

DE JONG, K. and SARMA, J. 1993. Generation Gaps Revisited *In:* Whitley, L.D. ed. *Foundations of Genetic Algorithms 2,* Morgan Kaufmann, also at www.gmu.edu

DEMING, W.E. 1950. *Some theory sampling.* New York/London: Wiley/Chapman and Hall.

DEMING, W.E. 1960. *Sample design in business research.* New York / London: Wiley.

DEMING, W.E. 1995. *Four Days with Dr. Deming.* Reading, Massahusetts: Addison-Wesley.

DOZIER, G., BOWEN, J. and BAHLER, D. 1994. Solving Small and Large Scale Constraint Satisfaction Problems Using a Heuristic-Based Microgenetic Algorithm. *Proceedings of the first IEEE Conference on Evolutionary Computation*, Orlando, Florida, USA. IEEE Neural Networks Council: IEEE.

FERNANDEZ, J. 2000. The GP Tutorial.

http://www.geneticprogramming.com/Tutorial/index.html (updated June 2000).

FISHER, R. A. 1925. *Statistical Methods for Research Workers.* Edinburgh: Oliver and Boyd.

FISHER, R. A. 1935. *The Design of Experiments.* Edinburgh: Oliver and Boyd.

FLOUDAS, C. A. and ANASTASIADIS, S. H. 1988. Synthesis of Distillation Sequences with Several Multicomponent Feed and Product Streams. *Chemical Engineering Science*, **43**(9), pp2407-2419.

FOGEL, D.B. 1994. An Introduction to Simulated Evolutionary Operation. *IEEE Transactions on Neural Networks*, **5**(1), pp3-14.

FOGEL, L. J., OWENS, A. J. and WALSH, M. J. 1966. *Artificial Intelligence through Simulated Evolution.* New York, USA: John Wiley and Sons.

FREENY, A. E. and NAIR, V. N. 1992. Robust Parameter Design with Uncontrolled Noise Variables. *Statistica Sinica.* **2**, pp. 313-334.

FRIESLEBEN, B. and HARTFELDER, M. 1993. Optimisation of Genetic Algorithms by Genetic Algorithms. *Artificial Neural Networks and Genetic Algorithms, Proceedings of the International Conference.* Innsbruck, Austria. Wein, Austria: Springer Verlag.

GEORGES-SCHLEUTER, M. 1992. Comparison of local mating strategies in massively parallel genetic algorithms *In:* MÄNNER, R. and MANDERICK, B., eds. *Parallel Problem Solving from Nature 2.* Amsterdam: North Holland. pp 553-562.

GOLDBERG, D.E. 1988(a). Genetic Algorithms and Rule Learning in Dynamic System Control. *Proceedings of the First International Conference on Genetic Algorithms and their Applications.* 24th July 1988, Carnegie-Mellon University, Pittsburgh, USA. Hillsdale, New Jersey, USA: Lawrence Erlbaum Associates. pp. 8-15.

GOLDBERG, D.E. 1988(b). *Genetic Algorithms in Search, Optimization and Machine Learning.* Reading, Ma., USA: Addison-Wesley.

GOLDBERG, D.E. 1989. Sizing Populations for serial and parallel genetic algorithms. *Proceedings of the third international conference on genetic algorithms.* San Mateo,Ca: Morgan Kaufmann. pp. 20-29.

GOLDBERG, D.E., DEB, K. and CLARK, J.H. 1991. Genetic Algorithms, Noise, and the Sizing of Populations. ILLiGAL Report No. 91010.

GREENALL, R. A. 1989. Taguchi Optimisation of the Manufacturing Process for an Injection Moulded Housing. *In*: BENDALL, A., DISNEY, J. and PRIDMORE, W. A., eds. *Taguchi Methods: Applications in World Industry.* Bedford, UK: IFS Publications, pp. 295-311.

HAHN, G. J. and DERSHOWITZ, A. F. 1974 Evolutionary Operation today - some survey results and observations. *Applied Statistics, The Journal of the Royal Statistical Society, Series C.* **23**(2), pp. 214-226.

HAMADA, M. 1990. Using Statistically Designed Experiments to improve Reliability and to achieve Robust Reliability. *Journal of Quality Technology,* **22**(1), pp. 38-45.

HAMADA, M. 1995. Using Statistically Designed Experiments to Improve Reliability and to Achieve Robust Reliability. *IEEE Transactions on Reliability.* **44**(2), pp. 206-215.

HEINSMAN, J. A. and MONTGOMERY, D. C. 1995. Optimization of a Household Product Formulation using a Mixture Experiment. *Quality Engineering.* **7**(3), pp. 583-600.

HEITTKÖTTER, J. and BEASLEY, D. 2000. Hitch Hiker's Guide to Evolutionary Computation, Issue 8.1, released 29 March 2000. http://www.cs.bham.ac.uk/Mirrors/ftp.de.uu.net/EC/clife.htm

HICKS, C. R. 1982. *Fundamental Concepts in the Design of Experiments.* 3rd edn. New York, London: Holt, Rinehart and Winston. pp. 37-38.

HOLLAND, J.H. 1992. *Adaptation in Natural and Artificial Systems: an* *introductory analysis with applications to biology, control and artificial intelligence.* First MIT Press edn. Cambridge, Ma., USA: MIT Press.

HUNTER, J. S. 1989. Statistical Quality Technology. *Industrial Quality and Productivity with Statistical Methods : A Joint Symposium of The Royal Society and The Royal Statistical Society.* 23rd March 1989. London. The Royal Society. pp. 597-604.

HUNTER, W. G. and KITTRELL, J. R. 1966. Evolutionary Operation: A Review. *Technometrics.* **8**, pp. 389-397.

HURLEY, P. 1994. Interactions: Ignore Them at Your Own Risk (How Taguchi's Confirmation Run Strategy Can Lead to Trouble). *Quality Engineering.* **6**(3), pp.451-457.

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. 1998. QS-9000: quality system requirements. Third edition. Carwin: Essex.

ISHIBUCHI,H., MURATA,T. 1996. Multi-Objective Genetic Local Search Algorithm. Proceedings of the IEEE International Conference on Evolutionary Computation. pp.119-124.

KACKER, R.N. and SHOEMAKER, A.C. 1986. Robust Design: A Cost-Effective Method For Improving Manufacturing Processes. *AT&T Technical Journal.* **65**(2), pp. 39-50.

KIDO, T., KITANO, H. and NAKANISHI, M. 1993. A Hybrid Search for Genetic Algorithms: combining genetic algorithms, TABU search and Simulated Annealing. *Proceedings of the fifth international conference on genetic algorithms.* p641.

KIRKPATRICK, S., GELATT, C. D. and VECCHI, M. P. 1983. Optimization by Simulated Annealing. *Science.* **220**, pp. 671-680.

KOZA, J.R. 1992. *Genetic Programming: on the programming of computers by means of natural selection.* Cambridge: MIT Press.

KRISHNAKUMAR, K. 1989. Micro-genetic Algorithms for Stationary and Non-stationary Function Optimization. *SPIE, Intelligent Control and Adaptive Systems.* **1195**, pp. 289-296.

KWONG, S., NG, A.C.L. and MAN, K. F. 1995. Improving Local Search in Genetic Algorithms for Numerical Global Optimization using Modified GRID-point Search Technique. *First IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications (GALESIA'95),* Halifax Hall, University of Sheffield, UK. London: IEE. pp. 419-423.

LANGDON, W.B. 1995. Pareto, Population Partitioning, Price and Genetic Programming.http://www.cs.ucl.ac.uk/staff/W.Langdon/WBL_papers.html#Langdon:1995:ppp

LAZAUSKAS, L. 1999. Test Functions and Benchmarks. www.maths.adelaide.edu.au/Applied/llazausk/alife/realfopt.htm. Last updated: 3rd March 1999.

LEE, S. and ROWLANDS, H. 1998. Reproduction in Genetic Algorithms using the Refocusing Operator. *Third International conference on Adaptive Computing in Design and Manufacture*. Plymouth, UK. pp 9-12.

LOUIS, S.J. 1997. Comparing Genetic and Other Search Algorithms. www.cs.unr.edu/~sushil/papers/thesis/thesishtml/node3.html

LOUIS, S.J., RAWLINS, G.J.E. 1993. Pareto Optimality, GA-easiness and Deception. (Indiana Uni) pp118-125 Proc. Fifth International Conference on Genetic Algorithms, S. Forrest (ed.), Morgan Kaufmann.

LOWE, C. W. 1964. Some Techniques of Evolutionary Operations. *Transactions of the Institute of Chemical Enginners*. **42**, pp. T334-T344.

LOWE, C. W. 1974. Evolutionary Operation in Action. *Applied Statistics, The Journal of the Royal Statistical Society, Series C*. **23**(2), pp. 218-226.

LUCAS, J.M. 1994. How to Achieve a Robust Process Using Response Surface Methodology. *Journal of Quality Technology.* **26**(4), pp. 248-260.

MASON, W.J., COVERSTONE-CARROLL, V., HARTMANN, J.W. 1998. Optimal Earth Orbiting Satellite Constellations via a Pareto Genetic Algorithm. AIAA/AAS Astrodyn. Spec. Conf., paper no. 98-4381 (Boston, Mass., Aug. 1998).

MCGOVERN, J.L. 1994(a). A Critique of the Taguchi Approach - Part I: A Presentation of Some Deficiencies and How These Limit Its Efficiency and Validity. *Journal of Coatings Technology.* **66**(830), pp. 65-70.

MCGOVERN, J.L. 1994(b). A Critique of the Taguchi Approach - Part I: An Alternative that is More Efficient. *Journal of Coatings Technology.* **66**(831), pp. 55-61.

MONTGOMERY, D. C. 1991. *Design and Analysis of Experiments.* 3rd edn. New York: J.Wiley and Sons.

MÜHLENBEIN, H., SCHOMISCH, M. and BORN, J. 1991. The Parallel Genetic Algorithm as a Function Optimizer. *Parallel Computing*, **17**, pp 619-632.

MURAKI, M., KATAOKA, K. and HAYAKAWA, T. 1986. Evolutionary Synthesis of a Multicomponent Multiproduct Separation Process. *Chemical Engineering Science.* **41**(7), pp. 1843-1851.

MURATA,T., ISHIBUCHI,H. 1995. MOGA:Multi-Objective Genetic Algorithms. Proceedings of the 2nd IEEE-ICEC International Conference on Evolutionary Computation. pp.289-294.

MUTH, J.F., THOMPSON, G.L.. 1963. *Industrial Scheduling.* Prentice-Hall: Englewood Cliffs, USA.

NACHTSHIEM, C. J., JOHNSON, P. E., KOTNOUR, K. D., MEYER, R. K. and ZUALKERNAN, I. A. 1990. Expert Systems for the Design of Experiments. *In:* GHOSH, S. ed. *Statistical design and analysis of industrial experiments.* New York: Marcel Dekker, pp. 109-131.

PARK, S.H. 1996. *Robust Design and Analysis for Quality Engineering.* Chapman and Hall: London.

PHADKE, M.S. 1986. Design Optimization Case Studies. *AT&T Technical Journal.* **65**(2), pp. 51-68.

PIGNATIELLO, J. J. and RAMBERG, J. S. 1991. Top Ten Triumphs and Tragedies of Genichi Taguchi. *Quality Engineering,* **4**(2), pp 211-225.

POLI, R. 1999. Evolutionary Computation Teaching at Birmingham. Congress on Evolutionary Computation. 6th-9th July 1999, Washington, USA. IEEE Piscataway.

PRINCIPIA CYBERNETICA WEB, 2000. Web Dictionary of Cybernetics and Systems: Pareto Optimality, http://pespmc1.vub.ac.be/ASC/PARETO_OPTIM.html

REEVES, C. R. 1993. Using Genetic Algorithms with Small Populations. *Proceedings of the fifth international conference on genetic algorithms*, 17th July 1993, Universtiy of Illinois at Urbana-Champaign, San Mateo, Ca: Morgan Kaufmann, pp. 92-99.

REEVES, C. R. and WRIGHT, C.C. 1995. Genetic Algorithms and Statistical Methods: a comparison. *First IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications (GALESIA),* 12th September 1995, Halifax Hall, University of Sheffield, UK. London: IEE.

RENCHENBERG, I. 1973. Evolutionsstrategie: Optimierung technischer systeme nach prinzipien der biolgischen evolution. Stuttgart: Frommann-Holtzboog Verlag

RENDERS, J-M. and BERSINI, H. 1994. Hybridizing Genetic Algorithms with Hill-Climbing Methods for Global Optimization: Two Possible Ways. *Proceedings of the first IEEE conference on evolutionary computation*; 27th June 1994, Orlando, Florida. NJ: Piscataway, pp 312-317.

ROY, R. 1990. A Primer on the Taguchi Method. New York: Van Nostrand Reinhold.

SAIT, S.M., YOUSSEF, H. 1999. *Iterative Computer Algorithms with Applications in Engineering: solving combinatorial problems.* Los Alamitos, Ca., USA : IEEE Computer Society.

SCHEFFÉ, H. 1958. Experiments with Mixtures. *Journal of the Royal Statistical Society B,*.**20**, pp. 344-360.

SCHNECKE, V. 1997. Genetic design of VLSI layouts. *In* ZALZALA, A.M.S. and FLEMING, P.J. eds. *Genetic algorithms in engineering systems.* London: IEE, pp229-253.

SCHWEFEL, H-P. 1965. Kybernetische evolution als strategie der experimentellen forschung in der strmungstechnik. Diploma thesis: Technical University of Berlin.

SCHWEFEL, H.-P. 1975. Bindre optimierung durch somatische mutation. Technical report: Working group of Bionic and Evolution Techniques at the Institute of Measurement and Control Technology of the Technical University of Berlin and the Central Animal Laboratory of the Medical High School of Hanover.

SHOEMAKER, A. C. and KACKER, R. N. 1988. A Methodology for Planning Experiments in Robust Product and Process Design. *Quality and Reliability Engineering International*, **4**(2), pp 95-103.

SPENDLEY, W., HEXT, G. R. and HIMSWORTH, F. R. 1962. Sequential

Application of Simplex Designs in Optimisation and Evolutionary Operation. *Technometrics,* **4**, pp. 441-461.

SRINIVAS, M. and PATNAIK, L. M. 1994. Genetic Algorithms: A Survey. *Computer*, June 1994 edition, pp. 17-26.

STARKWEATHER, T., WHITLEY, D. and MATHIAS, K. 1990. Optimization using distributed genetic algorithms. *Proceedings of Parallel Problem Solving from Nature 1, Lecture Notes in Computer Science No 496*, Springer-Verlag, pp. 176-185.

SYSWERDA, G. 1989. Uniform Crossover in Genetic Algorithms. *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 2-9.

TAGUCHI, G. 1987. *System of Experimental Design.* American Supplier Institute.

THEIRENS, D., SUYKENS, J., VANDEWALLE, J. and DE NOOR, B. 1993. Genetic Weight Optimisation of a Feeedfoward Neural Network Controller. *Artificial Neural Networks and Genetic Algorithms, Proceedings of the International Conference.* Innsbruck, Austria. Wein, Austria: Springer Verlag. pp. 658-663.

TRIBUS, M. and SZONYI, G. 1989. An Alternative View of the Taguchi Approach. *Quality Progress,* **22**(5), pp. 46-52.

UNAL, R. and DEAN, E. B. 1995. Design for Cost and Quality: the Robust Design

Approach. http://akao.larc.nasa.gov/pap/robdes/robdes.html 8th September 1995., also In: *Journal of Parametrics*, **11**(1).

VAN LAARHOVEN, P. J. M. and AARTS, E. H. L. 1987. *Simulated Annealing: Theory and Applications*. Dodrecht, Holland: D. Reidel Publishing Company.

VINING, G. G. and MYERS, R. H. 1990. Combining Taguchi and Response Surface Philosophies: a dual response approach. *Journal of Quality Technology,* **22**(1), pp. 38-45.

WIENHOLT, W. 1993. A Refined Gentic Algorithm for Parameter Optimization Problems. *Proceedings of the Fifth International Conference on Genetic Algorithms.*

WINTER, G., PERIAUX, J., GALAN, M., CUESTA, P. 1995. *Genetic Algorithms in Engineering and Computational Science*. Wiley: Chichester, New York.

YAMADA, T. and NAKANO, R. 1995. A Genetic Algorithm with Multi-step Crossover for Job-Shop Scheduling Problems. *Proceedings of Genetic Algorithms in Engineering Systems: Innovations and Applications*. London, IEE. pp. 146-151.

ZALZALA, A.M.S. and FLEMING, P.J. (eds). 1997. *Genetic algorithms in engineering systems*. London: IEE

9. BIBLIOGRAPHY

ABBATTISTA, F. and DALBIS, D. 1996. Improving the Genetic Algorithms by means of a Cooperative Model. *Second On-line Workshop on Evolutionary Computation,* http://www.bioele.nuee.nagoya-u.ac.jp/wec2/. 4-22 March 1996

ACKLEY, D. H. 1987. An Empirical Study of Bit Vector Optimization. *In:* DAVIS, L., ed. *Genetic Algorithms and Simulated Annealing.* London: Pitman.

ALBRECHT, R. F.; REEVES, C. R., and STEELE, N. C. 1993. Introduction. *Artificial Neural Networks and Genetic Algorithms, Proceedings of the International Conference. Innsbruck, Austria; 1993.* Wein, Austria: Springer Verlag.

AMIN, S. and FERNANDEZ-VILLACANAS, J. L. 1997. Dynamic local search. *GALESIA, 2nd IEE/IEEE International Conference on GA Applications.* 2nd –4th September, 1997. University of Strathclyde, Glasgow, UK.

ANDERSON, V. L. and MCLEAN, R. A. 1974. *Design of experiments: a realistic approach.* New York: Marcel Dekker.

anonymous. 1996. Design of Experiments For Processing Industries. http://www.sme.org/conf/quality/doeprdbr.html. (8th September 1996).

anonymous. 1996. Combinatorial and Real Function Optimisation. http://iridia.ulb.ac.be/projects/combi.html. (accessed 5th March 1996).

ANTHONY, J. and KAYE, M. 1998. Key Interactions. *Manufacturing Engineer.* June 1998. pp136-138.

BÄCK, T., HAMMEL, U. and SCHWEFEL, H-P. 1997. Evolutionary Computation: Comments on the History and Current State. *IEEE Transactions on Evolutionary Computation.* **1**(1), pp 3-17

BÄCK, T., RUDOLPH, G. and SCHWEFEL, H-P. Evolutionary Programming and Evolution strategies: Similarities and Differences. *The Second Annual Conference on Evolutionary Programming*; San Diego, California, USA. pp11-22.

BAKER, J. E. 1987 Reducing Bias and Inefficiency in the Selection Algorithm. *Proceedings of the Second International Conference on Genetic Algorithms.* 28th July 1987. MIT, USA. New Jersey, USA: Lawrence Erlbaum Associates, pp14-21.

BATES, A. 1995. Within your grasp: better plant performance and major savings. *Works Management.* **48**(9), pp16-17.

BERSINI, H. and SERONT, G. 1992. In Search of a Good Evolution-Optimization Crossover. *Personal Communication from Hugues Bersini.*

BERSINI, H. and VARELA, F. 1993. The Immune Learning Mechanisms: Reinforcement, Recruitment and their Applications. IRIDIA, Université Libre de Bruxelles Internal report: TR/IRIDIA/93-4. *Personal Communication from Hugues Bersini.*

BOX, G. E. P. 1966. Use and Abuse of Regression. *Technometrics.* **8**(4), pp625-629.

CARLSON, S.E., SHONKWILER, R. and MICHAEL, I. 1993. A Comparative Evaluation of Search Methods Applied to Catalog Selection. *Proceedings of the International Conference on Genetic Algorithms,* 17th July 1993, University of Urbana-Champaign USA. San Mateo, Ca. USA: Morgan Kaufmann. p630.

CHEN, W-H. and TIRUPATI, D. 1995. On-line Quality Management: Integration of Quality Inspection and Process Control. *Production and Operations Management,* **4**(3), pp242-262.

CHIPPERFIELD, A., FLEMING, P., POHLHEIM, H. and FONSECA, C. 1993. Genetic Algorithm Toolbox for use with Matlab User's Guide v1.2.

CLEVELAND, G.A. and SMITH, S.F. 1989. Using Genetic Algorithms to Schedule Flow Shop Releases. Proceedings of the International Conference on Genetic Algorithms 1989; pp160-169.

COWLEY, P. and PEARCE, R. 1997. Assessment of Applications for Optimisation using a Genetic Algorithm. *GALESIA, 2nd IEE/IEEE International Conference on GA Applications.* 2nd –4th September, 1997. University of Strathclyde, Glasgow, UK.

DE FALCO, I., DEL BALIO, R., DELLA CIOPPA, A. and TARANTINO, E. 1992. A Comparative Analysis of Evolutionary Algorithms for Function Optimisation. *WEC2 - 2nd Online Workshop on Evolutionary Computing*; 4th-22nd March 1992, http://www.bioele.nuee.nagoya-u.ac.jp/wec2/papers/p018.html; (3rd April 1996).

DE JONG, K. 1985. Genetic Algorithms: a 10 year perspective. *Proceedings of the First International Conference on Genetic Algorithms and their Applications,* 24th July 1985, Carnegie-Mellon University, Pittsburgh, USA. Hillsdale, New Jersey, USA: Lawrence Erlbaum Associates Publishers. pp 169-177.

DE JONG, K. and SPEARS, W. On the State of Evolutionary Computation: pp618-623.

DEHNAD, K. (ed). 1989. *Quality control, robust design, and the Taguchi method.* Pacific Grove, Calif. USA: Wadsworth & Brooks/Cole Advanced Books & Software.

DEMING, W.E. 1986. *Out of the crisis: quality, productivity and competitive position.* Cambridge: Cambridge University Press.

DIMOPOULOS, C. and ZALZALA, A.M.S. 2000. Recent Developments in Evolutionary Computation for Manufacturing Optimisation: problems, solutions and comparisons. www (accessed August 2000).

DORIGO, M., MANIEZZO, V. and COLORNI, A. 1996. The Ant System: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics - Part B.* **26**(1), pp1-13.

DOWSLAND, K. A. 1996. Genetic Algorithms - a tool for OR? *Journal of the Operational Research Society.* **47**, pp550-561.

EVOSTIM. 2000. The State of the Art in Evolutionary Approaches to Timetabling and Scheduling. www.dai.ed.ac.uk/daidb/people/homes/emmah/REPORT/draftsc/draftsc.html (accessed 1st September 2000).

FOGEL, D.B. 1991. *System Identification through Simulated Evolution: a Machine Learning Approach.* Needham Heights, Ma.,USA.: Ginn Press.

FOGEL, D.B. 1995. *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence.* Piscataway, NJ., USA: IEEE Press.

FUJIMOTO, H., LIAN-YI, C., TANIGAWA, Y. and IWAHASHI, K. 1995. FMS Scheduling by Hybrid Approaches using Genetic Algorithm and Simulation. *First IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications (GALESIA'95),* 12th-14th September 1995, Halifax Hall, University of Sheffield, UK. London: IEE. pp. 442 – 447.

GONZALEZ, B., TORRES, M., and MORENO, J. A. 1995. A Hybrid Genetic Algorithm Approach for the "No-Wait" Flow Shop Scheduling Problem. *Genetic Algorithms in Engineering Systems: Innovations and Applications.*

GREFENSTETTE, J.J. 1986. Optimization of Control Parameters for Genetic Algorithms. *IEEE Transactions on Systems, Man and Cybernetics,* **SMC-16**(1), pp. 122-128.

GREFENSTETTE, J.J. 1987. Genetic Algorithms and their Applications. *Proceedings of the Second International Conference on Genetic Algorithms.* 28th July 1987. MIT, Cambridge, USA. New Jersey, USA: Lawrence Erlbaum Associates.

HAJELA, P. and LIN, C-Y. 2000. Real Versus Binary Coding in Genetic Algorithms: A Comparative Study. *Computational Engineering using Metaphors from Nature.* Edinburgh: Civil-Comp Press. pp. 77-83.

HAMMERSTROM, D. 1993. Neural Networks at Work. *IEEE Spectrum.* June edition. pp.26-32.

HAYKIN, S.S. 1994. *Neural Networks: A Comprehensive Foundation.* New York, USA: MacMillan.

HEBB, D. O. 1949. *The organization of behaviour: a neuropsychological theory.* New York, London: Wiley.

HEITKÖTTER, J. and BEASLEY, D. 1995. *Hitch Hiker's ,Guide to Evolutionary Computation.* (Issue 3.4). (www release 11th December 1995)

HOPFIELD, J. J. 1982. Neural Networks and Physical Systems with Emergent Collective Computational Abilities. *Proceedings of the National Academy of Sciences.* **79**, pp. 2554-2558.

HOUCK, C.R., JOINES, J.A. and KAY, M.G. A Genetic Algorithm for Function Optimization: A Matlab implementation. www.ncsu.edu (accessed August 2000).

IMAM, M.H. and AL-SHIHIRI, M..A. 2000. A Primitive Crossover for Improving the Reliability of Genetic Algorithms for Structural Optimization. *Computational Engineering using Metaphors from Nature.* Edinburgh: Civil-Comp Press. pp. 91-97.

KEANE, A. J. 1993. Structural Design for Enhanced Noise Performance Using

Genetic Algorithm and other Optimization Techniques. *Artificial Neural Networks and Genetic Algorithms, Proceedings of the International Conference*. Innsbruck, Austria. Wein, Austria: Springer Verlag. pp. 536-543.

LANE, R. and OCHILTREE, B. C. 1989. Simplex Optiomisation Technique for Development of Rubber Formulations. *Plastics and Rubber International*. **14**, pp. 28-32.

LANGERMAN FALSE SWARZBERG, S., SERONT, G. and BERSINI, H. 1994. S.T.E.P.: The Easiest Way to Optiomize a Function. *Proceedings of the First IEEE Confernece on Evolutionary Computation*. Orlando, Florida, USA. IEEE. pp. 519-524.

LIPPMANN, R. P. 1987. An Introduction to Computing with Neural Nets. *IEEE ASSP Magazine*. April 1987 edition, pp. 4-22.

LINDFIELD, G. and JOHN, P. 1995. *Numerical Methods using Matlab*. Chichester: Ellis Horwood. pp 284-294.

LOGAN, B. and POLI, R. 1996. Route planning with GA*. *1st Online Workshop on Soft Computing*, 19th August 1996. http://www.bioele.nuee.nagoya-u.ac.jp/wsc1/.

MAHFOUD, S.W. Crossover Interactions Among Niches. *First IEEE Conference on Evolutionary Computation, World Congress on Computational Intelligence,* University of Illinois at Urbana-Champaign Internet report. pp. 188-193.

MAHFOUD, S.W. Genetic Drift in Sharing Methods. *First IEEE Conference on Evolutionary Computation, World Congress on Computational Intelligence*, University of Illinois at Urbana-Champaign Internet report. pp. 67-72.

MINSKY, M. L. and PAPERT, S. 1969. *Perceptrons: an Introduction to Computational Geometry.* Cambridge, Ma.,USA., London: MIT Press.

MITCHELL, M. 1996. *Recent Papers by Melanie Mitchell.* http://www.santefe.edu/~mm/paper-abstrats.html. (created: 1st November 1996).

MITCHELL, M. and HOLLAND, J.H. 1993. When Will a Genetic Algorithm Outperform Hill Climbing? *Proceedings of the fifth international conference on genetic algorithms.* and at http://www.santafe.edu/~mm/paper-abstracts.html#ga-hillc. (accessed 10th October 1996).

MCCULLOCH, W. S. and PITTS, W. A. 1943. Logical Calculus of the Ideas Imminent in Nervous Activity. *Bulletin of Mathematical Biophysics.* **5**, pp. 115-133.

PATTON, R. J., CHEN, J. and LIU, G. P. 1995. Robust Fault Detection of Dynamic Sytems via Genetic Algorithms. *First IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications (GALESIA'95),* Halifax Hall, University of Sheffield, UK. 12th September 1995.

London: IEE.

PHADKE, M.S. 1989. *Quality Engineering using Robust Design.* Englewood Cliffs, NJ, USA: P T R Prentice Hall.

PHAM, D. T. and OZTEMEL, E. 1995. An integrated neural network and expert system tool for statistical process control. *Part B, Journal of Engineering Manufacture*, **209**(B2), pp. 91-97.

POLI, R. and LOGAN, B. 1992. Evolutionary Computation Cookbook: Recipes for designing New Algorithms. *2nd Online Workshop on Evolutionary Computation*, 4th – 22nd March 1992, http://www.bioele.nuee.nagoya-u.ac.jp/wec2/papers/d020.html.

PRYDE, M. and ROWLANDS, H. 1997. GAEVO – Genetic Algorithms with Evolutionary Operation. *First European Conference on Intelligent Management Systems in Operations.* 25th – 26th March 1997. University of Salford, UK

REES, D.G. 1989. *Essential Statistics.* 2nd edn. London: Chapman and Hall.

REEVES, C. R. 1995. *Modern Heuristic Techniques For Combinatorial Problems.* London: McGraw-Hill.

RIBEIRO-FILHO, J. L., TREVLEAVEN, P. C. and ALIPPI, C. 1994. Genetic Algorithm Programming Environments. *Computer.* June1994 edition, pp. 28-43.

ROY, R. and PARMEE, I. C. 1996. Adaptive Restricted Tournament Selection and a Local Hill Climbing Hybrid for the Identification of Multiple "Good " Design Solutions. *Second On-line Workshop on Evolutionary Computing,* 4th –22nd March 1996, http://www.bioele.nuee.nagoya-u.ac.jp/wec2/

ROY, R. CAVE, P. and PARMEE, I. 1994. Artificial Neural Networks and Taguchi's Methodology to Model a Complex Non-Linear System. *Proceedings of ACEDC'94*, 1994; Plymouth, pp 114-116.

RUMMELHART, D. E., MCLELLAND, J. L. and THE PDP RESEACH GROUP. 1986(a). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition; Vol. 1: Foundations.* Cambridge, Ma., USA, London: MIT Press.

RUMMELHART, D. E., MCLELLAND, J. L. and THE PDP RESEACH GROUP. 1986(b). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition; Vol. 2: Psychological and Biological Models.* Cambridge, Ma., USA., London: MIT Press.

SINGH BAICHER, G. and TURTON, B.C.H. 2000. Comparative study for optimisation of causal IIR perfect reconstruction filter banks. *Congress on Evolutionary Computation 2000 (CEC00) volume 2,* 16th to 19th July 2000, La Jolla, Ca., USA. IEEE. pp. 974-977.

SMITH, J. and SUGIHARA, K. 1996. GA Toolkit on the Web. *1st Online*

Workshop on Soft Computing, 19th August 1996, http://www.bioele.nuee.nagoya-u.jp.ac/wsc1/

TEO, M-Y. and SIM, S-K. 1995. Training the Neocognitron Network Using Design of Experiments. *Artificial Intelligence in Engineering*, **9**, pp. 85-94.

THOMPSON, V. 1995. Process Management in Manufacturing. *Control Engineering Practice,* **3**(4), pp. 537-543.

WANG, G., GOODMAN, E.D. and PUNCH III, W.F. 1996. Simultaneous Multi-Level Evolution, *Second On-line Workshop on Evolutionary Computation,* 4th – 22nd March 1996. http://www.bioele.nuee.nagoya-u.ac.jp/wec2/

YAO, X. 1993. A Review of Evolutionary Artificial Neural Networks. *International Journal of Intelligent Systems*, **8**, pp. 539-567.

YEN, J., LEE, B. and LIAO, J.C. Using Fuzzy Logic and a Hybrid Genetic Algorithm for Metabolic Modelling. Texas A&M University Report.

YIP, P.P.C. and POA, Y-H. 1995. Combinatorial Optimization with Use of Guided Evolutionary Simulated Annealing. *IEEE Transactions on Neural Networks,* **6**(2), pp. 290-295.

YURET, D. and DE LA MAZA, M. 1993. Dynamic Hill Climbing: Overcoming the limitations of optimization techniques. *Second Turkish Symposium on Artificial Intelligence and Neural Networks*; pp 254-260.