Evolutionary Decomposition of Complex Design Spaces

by

CHRISTOPHER RICHARD BONHAM

A thesis submitted to the University of Plymouth
in partial fulfilment for the degree of

DOCTOR OF PHILOSOPHY

Plymouth Engineering Design Centre
School of Computing
Faculty of Technology

In collaboration with
British Aerospace plc, Warton
Rolls Royce plc, Filton

December 2000
Evolutionary Decomposition of Complex Design Spaces

Christopher Richard Bonham
BEng(Hons) MSc AMIMechE

ABSTRACT

This dissertation investigates the support of conceptual engineering design through the decomposition of multi-dimensional search spaces into regions of high performance. Such decomposition helps the designer identify optimal design directions by the elimination of infeasible or undesirable regions within the search space. Moreover, high levels of interaction between the designer and the model increases overall domain knowledge and significantly reduces uncertainty relating to the design task at hand.

The aim of the research is to develop the archetypal Cluster Oriented Genetic Algorithm (COGA) which achieves search space decomposition by using variable mutation (vmCOGA) to promote diverse search and an Adaptive Filter (AF) to extract solutions of high performance [Parmee 1996a, 1996b]. Since COGAs are primarily used to decompose design domains of unknown nature within a real-time environment, the elimination of apriori knowledge, speed and robustness are paramount. Furthermore COGA should promote the in-depth exploration of the entire search space, sampling all optima and the surrounding areas. Finally any proposed system should allow for trouble free integration within a Graphical User Interface environment.

The replacement of the variable mutation strategy with a number of algorithms which increase search space sampling are investigated. Utility is then increased by incorporating a control mechanism that maintains optimal performance by adapting each algorithm throughout search by means of a feedback measure based upon population convergence. Robustness is greatly improved by modifying the Adaptive Filter through the introduction of a process that ensures more accurate modelling of the evolving population.

The performance of each prospective algorithm is assessed upon a suite of two-dimensional test functions using a set of novel performance metrics. A six dimensional test function is also developed where the areas of high performance are explicitly known, thus allowing for evaluation under conditions of increased dimensionality. Further complexity is introduced by two real world models described by both continuous and discrete parameters. These relate to the design of conceptual airframes and cooling hole geometries within a gas turbine.

Results are promising and indicate significant improvement over the vmCOGA in terms of all desired criteria. This further supports the utilisation of COGA as a decision support tool during the conceptual phase of design.
This thesis is dedicated to three remarkable women: my Nan Nellie May Hubball and her immortal words, "Good, better, best"; Tarnya who provided the inspiration and sacrificed more than anyone else; and my mother, who over recent years has shown me what true courage is.

Christopher R. Bonham
Plymouth, England 2000
List of Contents

1. INTRODUCTION
   1.1. Project Aims ................................................................. 1
   1.2. Engineering Design ..................................................... 2
   1.3. Creative and Systematic Design ...................................... 4
   1.4. Computers and Design .................................................. 5
   1.5. Evolutionary Computation ............................................. 5
       1.5.1. Genetic Algorithms ............................................... 9
           1.5.1.1. Encoding ..................................................... 13
           1.5.1.2. Selection Strategies ....................................... 16
           1.5.1.3. Crossover Operators ....................................... 18
           1.5.1.4. Mutation Operator ......................................... 20
   1.6. Thesis Overview .......................................................... 21

2. CLUSTER ORIENTED GENETIC ALGORITHMS (COGAs)
   2.1. Introduction ............................................................... 23
       2.1.1. COGA and Multi-Modal Optimisation ....................... 26
           2.1.1.1. Crowding ...................................................... 26
           2.1.1.2. Sharing ........................................................ 27
           2.1.1.3. Restricted Tournament Selection ....................... 28
           2.1.1.4. Sequential Niching ......................................... 29
           2.1.1.5. Parallel Methods ............................................. 29
   2.2. Generating COGA Solutions ......................................... 30
       2.2.1. The Diverse Search Engine – Variable Mutation .......... 31
       2.2.2. The Adaptive Filter (AF) ........................................ 33
           2.2.2.1. Probability Density Functions ............................ 34
           2.2.2.2. The Normal Distribution .................................... 35
           2.2.2.3. Explicit Filtering ............................................. 37
           2.2.2.4. Implicit Filtering .............................................. 38
       2.2.3. The Final Clustering Set (FCS) ................................. 39
           2.2.3.1. The Modified Himmelblau Function ...................... 40
           2.2.3.2. Dynamic Growth of the FCS ............................... 41
           2.2.3.3. The Filtering Vector ......................................... 43
3. COGA AND THE SUPPORT OF CONCEPTUAL ENGINEERING DESIGN

3.1. Introduction ......................................................................................................... 46
   3.1.1. MiniCAPS .................................................................................................. 46
   3.1.2. TBCOM ................................................................................................. 47
3.2. Single Objective Decomposition: MiniCAPS Model ..................................... 48
3.4. Multi-Objective Decomposition: MiniCAPS Model ................................... 52
3.5. Infeasible Region Decomposition: TBCOM Model .................................. 55
3.6. The Graphical User Interface ........................................................................ 58
3.7. The Interactive Evolutionary Design Station ............................................ 62

4. IMPROVING THE PERFORMANCE OF THE DIVERSE SEARCH ENGINE

4.1. Introduction ......................................................................................................... 64
4.2. The Test Algorithms ........................................................................................... 65
   4.2.1. VmCOGA ............................................................................................... 66
   4.2.2. Modified Uniform Crossover and Incest Prevention (HUX/IP) ............. 66
   4.2.3. Deterministic Crowding ......................................................................... 67
   4.2.4. ECOGA ................................................................................................ 67
4.3. Performance Metrics ........................................................................................... 68
4.4. The 2d Test Functions ........................................................................................ 71
4.5. Results ................................................................................................................. 72
4.6. The Six Dimension Test Function .................................................................... 75
4.7. Application of Test Algorithms to 6d Function ........................................... 80
4.8. Hybridising the ECOGA and DC Algorithms ............................................. 82
4.9. Concluding Remarks ......................................................................................... 85

5. SEARCH SPACE SAMPLING AND THE EXPLORATION OF ISOLATED PEAKS

5.1. Introduction ......................................................................................................... 87
5.2. Halton Injection COGA (HiCOGA) .............................................................. 89
   5.2.1. Low Discrepancy Sequences ................................................................. 90
5.2.2. Transforming the Leaped Halton Sequence ........................................... 92
5.3. Spatial Selection COGA (SsCOGA) ............................................................... 94
  5.3.1. Hypersphere Density Metric ............................................................... 96
  5.3.2. Nearest Neighbour Euclidean Distance Metric ..................................... 97
5.4. Application of HiCOGA and SsCOGA to 6d Test Function ..................... 98
  5.4.1. Presentation of Results ......................................................................... 99
  5.4.2. Results ................................................................................................ 100
5.5. Concluding Remarks .................................................................................. 104

6. ADAPTIVE CONTROL

6.1. Introduction .................................................................................................... 106
6.2. Adaptive Genetic Algorithms ....................................................................... 107
6.3. The Feedback Measure – Population Convergence ..................................... 109
  6.3.1. Calculating the Convergence Rate .......................................................... 113
6.4. The Control Mechanism ............................................................................... 114
  6.4.1. Defining the Maximum Target Rate Error ............................................. 117
  6.4.2. The Adaptive Algorithms ..................................................................... 118
6.5. Target Convergence Rate Study .................................................................. 119
  6.5.1. Target Rate = \( \dot{\hat{C}}_{\text{vm}} \) .................................................. 120
  6.5.2. Target Rate = 1.5 \( \dot{\hat{C}}_{\text{vm}} \) ....................................................... 122
  6.5.3. Target Rate = 2 \( \dot{\hat{C}}_{\text{vm}} \) ......................................................... 124
6.6. Application to Additional Test Functions: The Calibration Curves ........... 126
  6.6.1. Application to TF1 and TF2 ................................................................ 127
  6.6.2. Application the miniCAPS, TBCOM and Keane’s Function ................. 133
6.7. Concluding Remarks .................................................................................. 135

7. THE DYNAMIC ADAPTIVE FILTER

7.1. Introduction .................................................................................................... 137
  7.1.1. Model Drift ......................................................................................... 138
  7.1.2. Model Mismatch .................................................................................. 138
    7.1.2.1. Kurtosis ..................................................................................... 138
    7.1.2.2. Skewness ..................................................................................... 139
7.2. The Probability Density Function Library ............................................... 139
7.2.1. Uniform Distribution .............................................................. 140
7.2.2. Exponential Distribution.......................................................... 140
7.2.3. Normal Distribution ................................................................. 141
7.2.4. Lognormal Distribution ............................................................ 141
7.2.5. Weibull Distribution ................................................................. 142
7.3. Closeness of Fit Metrics ............................................................... 143
7.3.1. Chi-Squared ($\chi^2$) Metric ....................................................... 143
7.3.2. Kolmogorov-Smirnov Metric ..................................................... 144
7.4. Application of PDFs to Multi-Dimensionality Test Functions ....... 145
7.4.1. DeJong F5 ................................................................................. 146
7.4.2. Michalewicz’s Function ............................................................ 147
7.4.3. MiniCAPS (Maximising SEP2) .................................................. 148
7.4.4. Keane’s 20d Bump Function ...................................................... 149
7.4.5. TBCOM ....................................................................................... 150
7.5. Filtering Threshold Probability Mapping ...................................... 152
7.6. Assessing the Performance of the DAF ....................................... 153
7.7. Integrating the DAF with SsCOGA and mssCOGA ...................... 155
7.8. Concluding Remarks ................................................................... 158

8. KNOWLEDGE DISCOVERY WITHIN THE FCS
8.1. Introduction .................................................................................. 159
8.2. Visualisation ................................................................................ 160
8.2.1. One dimensional (Univariate) Visualisation ................................ 162
8.2.2. Two Dimensional (Bivariate) Visualisation ............................... 167
8.2.2.1. Visualising Design Attributes ............................................. 170
8.2.3. Three Dimensional (Trivariate) Visualisation ............................ 173
8.2.4. Hypervariate and Advanced Visualisation Techniques ................ 177
8.3. Concluding Remarks ................................................................... 181

9. CONCLUSIONS
9.1. Conclusions ................................................................................ 183
9.2. Future Research Directions ......................................................... 187
Appendix A: Test Function 2
Appendix B: 6d Test Function – Plane 1
Appendix C: 6d Test Function – Plane 2
Appendix D: 6d Test Function – Plane 3
Appendix E: Keane’s 20d Bump Function
Appendix F: Hastings approximation to the standardised normal distribution
Appendix G: Approximating $\alpha$ and $\theta$ for the Weibull PDF

References
List of Tables

1.1 Definition of terms ................................................................................................... 12

2.1 Summary of Explicit and Implicit filtering .............................................................. 25

3.1 MiniCAPS Inputs ..................................................................................................... 47
3.2 MiniCAPS Outputs .................................................................................................. 47
3.3 TBCOM Inputs ....................................................................................................... 48
3.4 TBCOM Outputs ..................................................................................................... 48
3.5 Final clustering set composition ............................................................................ 50
3.6 TBCOM Constraint Model ................................................................................... 56

4.1 Performance metrics ................................................................................................. 69
4.2 Set Cover and Set Cover Sensitivity ........................................................................ 73
4.3 GDE and GDE Sensitivity ....................................................................................... 73
4.4 FCSave and FCSbest ............................................................................................. 74
4.5 The global high performance regions (HP = High performance) ......................... 80
4.6 FCS results for DC and the 30x30 ECOGA (values in parenthesis are standard deviations expressed as a % of measured value) ...................................................... 80
4.7 FCS results for the DC / ECOGA hybrid (the ratio in the first column relates to the proportion of search given to the respective algorithm, values in parenthesis are standard deviations expressed as a % of measured value) ...................................................... 83
4.8 FCS results for the ECOGA / DC hybrid (the ratio in the first column relates to the proportion of search given to the respective algorithm, values in parenthesis are standard deviations expressed as a % of measured value) ...................................................... 84
4.9 FCS results for the 1:3:1 DC/ECOGA/DC hybrid (the ratio in the first column relates to the proportion of search given to the respective algorithm, values in parenthesis are standard deviations expressed as a % of measured value) ...................................................... 85

5.1 A sample Halton sequence of ten points in five dimensions (m = 10, n = 5) .......... 91
5.2 Regional sensitivity rank ........................................................................................ 100
5.3 The test algorithms ............................................................................................... 100
5.4 FCS results for SsCOGA1,2 and HiCOGA1,2 (values in parenthesis are standard deviations expressed as a % of measured value) ...................................................... 101
5.5 FCS results for SsCOGA\textsubscript{3,4} and \textit{s} (values in parenthesis are standard deviations expressed as a \% of measured value) ......................................................... 103

6.1 The search profiles .................................................................................................. 111
6.2 The adaptive algorithms ......................................................................................... 118
6.3 The control parameter values in the generations preceding and succeeding the commencement of control, \(p(\text{mut})^*, n_{\text{Hi}}^*, n_{\text{ss}}^*\) indicate adapted parameters (\(n = 6\), section 6.3.1) .............................................................................................. 119
6.4 FCS results (\(\hat{C}_m^i = \hat{C}_m^\text{vm}\)) (values in parenthesis are standard deviations expressed as a \% of measured value) ............................................................................... 120
6.5 Relative RD (\(\hat{C}_m^i = \hat{C}_m^\text{vm}\)) (Low, medium and high relate to sensitivity groupings).................................................................................................................. 122
6.6 FCS results (\(\hat{C}_m^i = 1.5\hat{C}_m^\text{vm}\)) (values in parenthesis are standard deviations expressed as a \% of measured value) ............................................................................... 122
6.7 Relative RD (\(\hat{C}_m^i = 1.5\hat{C}_m^\text{vm}\)) (Low, medium and high relate to sensitivity groupings).................................................................................................................. 124
6.8 FCS results (\(\hat{C}_m^i = 2\hat{C}_m^\text{vm}\)) (values in parenthesis are standard deviations expressed as a \% of measured value) ............................................................................... 124
6.9 Relative RD (\(\hat{C}_m^i = 2\hat{C}_m^\text{vm}\)) (Low, medium and high relate to sensitivity groupings).................................................................................................................. 125
6.10 Regional sensitivity ranking TF1 and TF2 ............................................................. 128
6.11 FCS results for TF1 study (values in parenthesis are standard deviations expressed as a \% of measured value) ............................................................................... 128
6.12 FCS characteristics for TF2 study (values in parenthesis are standard deviations expressed as a \% of measured value) ............................................................................... 128
6.13 Relative Set cover TF1 and TF2 .......................................................................... 130
6.14 Average FCS fitness (mini\text{CAPS} is set to maximise ferry range) (\(\hat{C}_m^i = \hat{C}_m^\text{vm}\)) .................................................................................. 133
6.15 Relative set cover (mini\text{CAPS} is set to maximise ferry range) (\(\hat{C}_m^i = \hat{C}_m^\text{vm}\)) .................................................................................. 134

7.1 FCS results for \textit{vmCOGA} using the DAF (values in parenthesis are standard deviations expressed as a \% of measured value) ......................................................... 154
7.2 FCS results for SsCOGA and mssCOGA using the DAF (\(\hat{C}_m^i = 1.5\hat{C}_m^\text{vm}\)) (values in parenthesis are standard deviations expressed as a \% of measured value) .......... 156
7.3 RD results for SsCOGA and msCOGA (relaxed filtering vector) using the DAF (Low, medium and high relate to sensitivity groupings) .......................................... 157

8.1 Number of design variable combinations ........................................................................................................ 162

8.2 Correlation analysis for TBCOM Geometry 1 (Results are ranked in descending absolute magnitude) ........................................................................................................ 168

8.3 Correlation coefficient analysis for miniCAPS – maximising Range, SEP1 and ATR1 (results are ranked in descending absolute magnitude) .................................................................................................................................................................................... 169
List of Figures

1.1 The design process ................................................................................................... 3
1.2 Evolutionary computing paradigm ........................................................................... 7
1.3 The structure of a GP program (the nodes are represented by circles) .......... 9
1.4 The canonical or tripartite GA ................................................................................ 11
1.5 Floating point representation ................................................................................ 13
1.6 Binary representation .......................................................................................... 14
1.7 The structured chromosome (the highlighted variables indicate active areas within the chromosome, $d_i$, $x_i$, $y_i$ and $z_i$ represent binary segments relating to each variable) ................................................................. 16
1.8 Roulette wheel selection (population size of 10) ............................................... 17
1.9 Stochastic universal sampling (population size of 10) ....................................... 17
1.10 One point crossover .......................................................................................... 18
1.11 Failure of one point crossover .......................................................................... 18
1.12 Two point crossover ......................................................................................... 19
1.13 Success of two point crossover ......................................................................... 19
1.14 Uniform crossover ............................................................................................ 19
1.15 Double point reduced surrogate crossover ....................................................... 19
1.16 The action of mutation ..................................................................................... 20

2.1 The COGA mechanism ....................................................................................... 25
2.2 Expansion of “Generate Solutions” block ............................................................ 32
2.3 A Probability Density Function ............................................................................ 35
2.4 Normal distribution ($\mu = 0.75$, $\sigma = 0.25$) $n$ = number of observations $X$ is a random variable of the range $x_{\min} \leq X \leq x_{\max}$ ................................................................. 36
2.5 The standardised normal distribution ($\mu = 0$, $\sigma = 1$) .................................. 37
2.6 The action of the adaptive filter ....................................................................... 38
2.7 Surface plot of the modified Himmelblau function ............................................ 40
2.8 Contour plot of the modified Himmelblau function .......................................... 40
2.9 FCS after search stage 1 .................................................................................. 41
2.10 FCS after search stage 2 .................................................................................. 41
2.11 FCS after search stage 3 .................................................................................. 42
2.12 FCS after search stage 4 .................................................................................. 42
2.13 FCS after search stage 5 (Colour key is common to all four plots) ................. 42
2.14 Surface plot of the seven peak function ................................................................. 43
2.15 Contour plot of the seven peak function (All contours below 0.5 omitted) .......... 43
2.16 \( RF = \{0.0, 0.0, 0.0, 0.0, 0.0\} \) ........................................................................ 44
2.17 \( RF = \{0.5, 0.5, 0.5, 0.5, 0.5\} \) ........................................................................ 44
2.18 \( RF = \{1.0, 1.0, 1.0, 1.0, 1.0\} \) ........................................................................ 44
2.19 \( RF = \{1.5, 1.5, 1.5, 1.5, 1.5\} \) (Colour key is common to all four plots) .......... 44

3.1 Minimising Landing Speed \( RF = \{1.0, 1.0, 1.0, 1.0, 1.0\} \) ................................ 49
3.2 Minimising Landing Speed \( RF = \{1.5, 1.5, 1.5, 1.5, 1.5\} \) ................................ 49
3.3 Minimising Landing Speed \( RF = \{2.0, 2.0, 2.0, 2.0, 2.0\} \) ................................ 49
3.4 Minimising Landing Speed \( RF = \{2.5, 2.5, 2.5, 2.5, 2.5\} \) (Colour key is common to all four plots) ................................................................. 49
3.5 FCS taken from a COGA run upon the TBCOM model ........................................ 51
3.6 Idealised high performance boundaries for the results shown in figure 3.5 ............ 51
3.7 Maximising Ferry Range \( RF_{\text{Range}} = \{1.0, 1.0, 1.0, 1.0, 1.0\} \) .............. 52
3.8 Maximising ATR \( RF_{\text{ATR}} = \{1.0, 1.0, 1.0, 1.0, 1.0\} \) ................................... 52
3.9 Maximising SEP \( RF_{\text{SEP}} = \{1.0, 1.0, 1.0, 1.0, 1.0\} \) ...................................... 53
3.10 Compromise region Range \( \cap \) ATR \( RF_{\text{Range}} = \{1.0, 1.0, 1.0, 1.0, 1.0\} \) \( RF_{\text{ATR}} = \{1.0, 1.0, 1.0, 1.0, 1.0\} \) \( RF_{\text{SEP}} = \{1.0, 1.0, 1.0, 1.0, 1.0\} \) ........................................... 54
3.11 Compromise region Range \( \cap \) ATR \( RF_{\text{Range}} = \{1.0, 1.0, 1.0, 1.0, 1.0\} \) \( RF_{\text{ATR}} = \{1.0, 1.0, 1.0, 1.0, 1.0\} \) \( RF_{\text{SEP}} = \{0.5, 0.5, 0.5, 0.5, 0.5\} \) .......................................... 54
3.12 Compromise region SEP \( \cap \) ATR \( RF_{\text{Range}} = \{1.0, 1.0, 1.0, 1.0, 1.0\} \) \( RF_{\text{ATR}} = \{1.0, 1.0, 1.0, 1.0, 1.0\} \) \( RF_{\text{SEP}} = \{0.0, 0.0, 0.0, 0.0, 0.0\} \) .......................................... 54
3.13 Compromise region Range \( \cap \) SEP \( RF_{\text{Range}} = \{1.0, 1.0, 1.0, 1.0, 1.0\} \) \( RF_{\text{ATR}} = \{1.0, 1.0, 1.0, 1.0, 1.0\} \) \( RF_{\text{SEP}} = \{0.0, 0.0, 0.0, 0.0, 0.0\} \) .......................................... 54
3.14 Compromise region SEP \( \cap \) ATR \( \cap \) Range \( RF_{\text{Range}} = \{1.0, 1.0, 1.0, 1.0, 1.0\} \) \( RF_{\text{ATR}} = \{1.0, 1.0, 1.0, 1.0, 1.0\} \) \( RF_{\text{SEP}} = \{0.0, 0.0, 0.0, 0.0, 0.0\} \) .......................................... 55
3.15 Minimise Wcr (Subject to the constraint equations) \( RF = \{0.0, 0.0, 0.0, 0.0, 0.0\} \) .... 57
3.16 Maximising constraint violation 1 ........................................................................ 58
3.17 Maximising constraint violation 3 ........................................................................ 58
3.18 Defining the search space and optimised objective ............................................. 59
3.19 Defining the COGA parameters \( g \), \( m \) and \( RF \) ..................................................... 59
3.20 FCS analysis presented at the end of search ......................................................... 60
3.21 Warning displayed when no solutions pass the filtering process ....................... 60
3.22 Hyperplane selection ......................................................................................... 60

xii
3.23 2d scatter plot showing Wing Aspect Ratio and Gross Wing Plan Area ............... 61
3.24 Examining individual design solutions ...................................................................... 61
3.25 The Interactive Evolutionary Design Station (IEDS) ............................................. 63

4.1 Test function 1 (TF 1) .............................................................................................. 71
4.2 Test function 1 high performance regions ................................................................. 71
4.3 Test function 2 (TF 2) .............................................................................................. 72
4.4 Test function 2 high performance regions ................................................................. 72
4.5 Regional decomposition results for test function 1 .................................................. 74
4.6 Regional decomposition results for test function 2 .................................................. 75
4.7 Plane 1 (no added noise) ......................................................................................... 78
4.8 Plane 1 high performance regions ............................................................................ 78
4.9 Plane 2 (no added noise) ......................................................................................... 78
4.10 Plane 2 high performance regions .......................................................................... 78
4.11 Plane 3 (no added noise) ......................................................................................... 78
4.12 Plane 3 high performance regions ............................................................................ 78
4.13 Regional decomposition results for the 6d test function ......................................... 81
4.14 Regional decomposition results for the 3:1:3 ECOGA / DC / ECOGA hybrid ........ 85

5.1 The reproduction phase in hiCOGA \( (n_{Hi} = \text{number of Halton injections} / \text{generation} n_{cross} = \text{number of reproduced individuals} / \text{generation}) \) ...................................... 90
5.2 Transforming the LH sequence ................................................................................. 92
5.3 A 60 point LH sequence represented in a 2-dimensional hyperplane (The centroid of the hyperplane is indicated by the chequered circle) ........................................... 93
5.4 A transformed 60 point LH sequence represented in a 2-dimensional hyperplane (The transformed centroid of the hyperplane is indicated by the chequered circle) ................................................................. 93
5.5 Effect of increasing population size \( m (n = 2) \) ......................................................... 94
5.6 Effect of increasing dimensionality \( n (m = 50) \) ....................................................... 94
5.7 The reproduction phase in SsCOGA \( (n_{ss} = \text{number of spatial selections} / \text{generation} n_{cross} = \text{number of reproduced individuals} / \text{generation}) \) ..................................................... 95
5.8 A snapshot of an evolving population (modified Himmelblau function) ................. 96
5.9 The hypersphere local density measure ................................................................. 96
5.10 The nearest neighbour Euclidean distance local density measure ......................... 98
5.11 Regional sensitivity classification (vmCOGA) ........................................................ 99
5.12 Relative regional decomposition results for SsCOGA\(_1\) and SsCOGA\(_2\) .................. 102
5.13 Relative regional decomposition results for HiCOGA_1 and HiCOGA_2 ........................................ 102
5.14 Relative regional decomposition results for SsCOGA_3, SsCOGA_4 and SsCOGA_5 .......................................................... 104

6.1 Open loop COGA .................................................................................................................................................. 106
6.2 Closed loop COGA ...................................................................................................................................... 106
6.3 Search profiles TF2 (2 dimensions) ................................................................................................................... 112
6.4 Search profiles 6d function (6 dimensions) ...................................................................................................... 112
6.5 Search profiles TBCOM – Geometry 3 (11 dimensions) ...................................................................................... 112
6.6 Search profiles Keane’s bump function (20 dimensions) .................................................................................... 112
6.7 Calculating C_m ............................................................................................................................................. 113
6.8 Setting the number of regression points ........................................................................................................ 114
6.9 Determining the control gain .......................................................................................................................... 116
6.10 Determining the maximum expected error (Δ_{max}) ..................................................................................... 118
6.11 Calculating C_m for vmCOGA on the 6d test function ....................................................................................... 120
6.12 Relative regional decomposition results (C_m = C_{vm}) ............................................................................... 121
6.13 Relative regional decomposition results (C_m = 1.5C_{vm}) ........................................................................... 123
6.14 Relative regional decomposition results (C_m = 2C_{vm}) .............................................................................. 125
6.15 vmCOGA calibration curves .......................................................................................................................... 126
6.16 Regional decomposition results for TF1 (C_m = C_{vm}) .................................................................................. 129
6.17 Regional decomposition results for TF1(C_m = 1.5C_{vm}) ................................................................................ 129
6.18 Regional decomposition results for TF2 (C_m = C_{vm}) .................................................................................. 130
6.19 Regional decomposition results for TF2 (C_m = 1.5C_{vm}) ........................................................................... 130
6.20 Relative proportion of all HP solutions for TF1 (C_m = C_{vm}) ....................................................................... 132
6.21 Relative proportion of all HP solutions for TF1 (C_m = 1.5C_{vm}) ................................................................. 132
6.22 Relative proportion of all HP solutions for TF2 (C_m = C_{vm}) ....................................................................... 132
6.23 Relative proportion of all HP solutions for TF2 (C_m = 1.5C_{vm}) ................................................................. 132

7.1 Kurtosis of the normal distribution .................................................................................................................. 139
7.2 Skewness of the normal distribution ............................................................................................................. 139
7.3 The uniform distribution (a = 0.25, b = 1.25) ............................................................................................. 140
7.4 The exponential distribution (μ = 0.75) ........................................................................................................... 141
7.5 The normal distribution ($\mu = 0.75, \sigma = 0.25$) ........................................................... 141
7.6 Lognormal distribution ($\mu = -0.75, \sigma = 0.4$) ........................................................... 142
7.7 The Weibull distribution ($\alpha = 6.2, \theta = 1.2$) ............................................................... 143
7.8 Application of library PDFs to DeJong's F5 function ...................................................... 146
7.9 Application of library PDFs to Michalewicz's function .................................................... 147
7.10 Application of library PDFs to miniCAPS (minimising SEP2) ....................................... 148
7.11 Application of library PDFs to Keane's 20d Bump function (uniform and exponential curves omitted) ..................................................................................... 149
7.12 Application of library PDFs to TBCOM (vmCOGA search) ........................................... 150
7.13 Application of library PDFs to TBCOM (Premature convergence) ................................. 150
7.14 Application of library PDFs to TBCOM (Normal search) ............................................. 150
7.15 Application of library PDFs to TBCOM (Random search) ............................................. 150
7.16 Calculating $P(\text{Pass}_{RF})$ from a standardised normal distribution ....................... 153
7.17 Calculating $F_{RF}$ for any PDF ..................................................................................... 153
7.18 Regional decomposition results (given as a percentage increase or decrease of the control filter) ....................................................................................................... 155
7.19 Regional decomposition results for SsCOGA (relaxed filter) using the DAF and the relaxed filtering vector ........................................................................................ 156
7.20 Regional decomposition results for mssCOGA using the DAF and the relaxed filtering vector .......................................................................................................... 157

8.1 Number of solutions within each group ........................................................................ 163
8.2 Average fitness of the solutions within each group ....................................................... 163
8.3 Standard deviation of fitness within each group ............................................................. 163
8.4 Number of solutions within each group ........................................................................ 163
8.5 Average fitness of the solutions within each group ....................................................... 163
8.6 Standard deviation of fitness within each group ............................................................. 163
8.7 A sample box plot ........................................................................................................ 164
8.8 Box plot of fitness relating to each geometry configuration ......................................... 164
8.9 Box plot of $F_f$ for each geometry configuration ......................................................... 165
8.10 Box plot of $R_s$ for each geometry configuration ......................................................... 165
8.11 Non-parametric PDF for wing sweep when ATR1, STR1 and take-off mass are optimised ........................................................................................................................................ 166
8.12 Scatterplot matrix (TBCOM - Geometry 2) ............................................................. 167
8.13 Design variables with a high correlation ...................................................................... 169
8.14 Design variables with a low correlation ................................................................. 169
8.15 Design attribute (fitness) represented by colour .................................................... 171
8.16 Design attribute (ATR1) represented by colour ..................................................... 171
8.17 Design attribute (ATR2) represented by colour ..................................................... 171
8.18 Design attribute (fitness) represented by size ....................................................... 171
8.19 Design attribute (fitness) represented by surface plot ........................................... 172
8.20 Design attribute (fitness) represented by contour plot .......................................... 172
8.21 Design attribute (fitness) represented by interpolated colourmap Colour key is common with figure 8.15 ......................................................................................... 173
8.22 Wing sweep and t / c ratio represented within a hyperplane, point colour represents aspect ratio - single colour representation ................................................................. 174
8.23 Wing sweep and t / c ratio represented within a hyperplane, point size represents aspect ratio, point colour represents design attribute (fitness) .............................. 174
8.24 Wing sweep and thickness / chord ratio represented within a hyperplane and conditioned against aspect ratio, point colour represents design attribute (fitness) - two colour representation .............................................................................. 175
8.25 Thickness / chord ratio, wing sweep and aspect ratio represented within a hypercube, point colour represents design attribute (fitness) – spectral colour representation ...................................................................................................................... 176
8.26 Hypercube rotated 90° about Aspect ratio axis .................................................... 176
8.27 Hypercube rotated 180° about Aspect ratio axis .................................................... 176
8.28 Hypercube rotated 270° about Aspect ratio axis .................................................... 176
8.29 Wing sweep, wing area and thickness / chord ratio represented within a hypercube and conditioned against cruise height, point colour represents design attribute (fitness) - two colour representation .......................................................... 177
8.30 Linked hyperplane with brushed points (wing area vs. cruise height) .................... 178
8.31 Linked hyperplane with brushed points (t / c ratio vs. wing sweep)......................... 178
8.32 Chernoff face representing six design variables all at their minimum value ............. 179
8.33 Chernoff face representing six design variables all at their mid-point value ............. 179
8.34 Chernoff face representing six design variables all at their maximum value .......... 179
8.35 6d world-within-world representation ...................................................................... 180
8.36 Distortion applied progressively to a hypercube, each individual cube represents an equal sample of the space in the original undistorted configuration. The focal point is located at the centroid (reproduced with permission S. Carpendale © 1997 IEEE) ........................................................................................................... 181
Acknowledgements

I will be forever indebted to my father, mother and sister who throughout my life have given me the belief and encouragement to succeed and the support and sympathy when I failed.

It seems fitting that a thesis with its premise drawn from Darwin's origin of the spices should be researched and written in Plymouth, from where Darwin set sail in 1832 for his voyage on HMS Beagle. Almost a century and a half after the publication of his work, I wish to thank the people of Plymouth Engineering Design Centre past and present who continue to implement his ideas, albeit with the help of the silicon chip.

On behalf of Orange Group people everywhere, final acknowledgement is given to those who said I didn't have the ability, inclination or dedication.
Author's Declaration

At no time during the registration for the degree of Doctor of Philosophy has the author been registered for any other University award.

This study was financed by the EPSRC, Rolls Royce plc and the University of Plymouth.

Throughout the course of the research, close links were maintained with Rolls Royce plc, Filton and British Aerospace plc, Warton. The research has been presented at the following international conferences.

Publications:


Signed ...................................................

Date 1/12/2000 ...............................................

xix
CHAPTER 1

Introduction

1.1 Project Aims

The research presented in the following pages relates to the continued development of the Cluster Oriented Genetic Algorithm (COGA) approach, initially developed by Parmee [Parmee 1996a & 1996b] to support the designer during the conceptual stages of the design process (Chapter 2).

The four main aims of the research project may be summarised as:

1. Propose methodologies for the use of COGA to support conceptual design within domains described by real-world system models of differing dimensionality. Develop a Graphical User Interface (GUI) which integrates COGA into the conceptual design environment (Chapter 3).

2. Produce a test function and corresponding series of performance measures that tests the capabilities of COGA and its successors (Chapter 4). The test function should be developed in such as way as to eliminate the need for apriori knowledge relating the domain it proscribes. The performance measures should be sufficiently rigorous to test each of COGA operating requirements outlined in section 2.1

3. Improve the performance, reliability and ease of use of the archetypal COGA algorithm (Chapters 4, 5, 6 and 7).

4. Suggest methodologies for presenting the results from a COGA run in a manner that maximises presented information, without overburdening or overloading the designer or design team (Chapter 8).
Since the raison d'être of COGAs is the support of the design process, this thesis begins with a review of engineering design, its constituent processes and the role computers play during the conceptual stages of design.

### 1.2 Engineering Design

Defining design is not a simple matter. Ruckert [Ruckert et al. 1997] suggests that design is the process of moving from an unwanted initial stage to a desired goal state through a barrier that prevents the goal state from being achieved. French [French 1985] argues that design is all the processes that encompass conception, innovation, visualisation, calculation and refinement, whilst Naveiro [Naveiro 1997] states that design is a process of permanent innovation by means of multi-disciplinary actions of technological, economic, social, cultural and environmental characteristics.

To simplify the design process one may break it down into a number of interrelated stages (figure 1.1). All design activity begins with a need, this may be the development of a new product or the improvement of an existing one. Closely following from this is a clear definition of the problem. This definition should include not only a statement of the design task but also any constraints that are placed upon the final design (completion date, cost etc.). In the conceptual design phase the designer formulates general concepts of how to solve the problem. The lack of any embodiment or detailed design studies at this point makes conceptual design low risk, since the design may be changed without any significant effect upon either the cost or the duration of the design process. However, conceptual design places heavy demands upon the designer since not only are decisions often made with incomplete, abstract or inconsistent information relating to the design domain [Ullman et al. 1995], but there is also a requirement for significant knowledge relating to all factors affecting the design, (performance, economical, aesthetic, socio-economic etc.). Consequently, a design team comprised of members from a broad expert base is often used
where the designers may share knowledge [Richer et. al. 199]. Once the general concepts have been developed and evaluated, embodiment commences. This involves further development of the general concepts. Throughout this phase the uncertainty relating to the design reduces (i.e. the designer knows roughly what the design looks like). During this stage interaction with conceptual design is often significant, since knowledge gained at the embodiment stage may require revision of the initial design concepts. Furthermore, it is rare that the ideas developed at the conceptual stage perform well in all the criteria defined during the analysis of the problem [Dixon 1966]. The final stage of figure 1.1, is detailed design. Detailed design involves the development of the final design by making small changes to the design developed during the embodiment stage. Consequently, detailing should only commence when the majority of the design has been fixed and confidence relating to it is high, the feedback loop of figure 1.1 therefore represents a highly undesirable return to and earlier design phase.

Figure 1.1: The design process
1.3 Creative and Systematic Design

It is widely accepted that there are two distinct design processes, creative and systematic. Creative design is the ability to produce new ideas and solutions without the use of preconceived ideas, whereas systematic design is the following of preconceived ideas or logical steps [Haapasalo et. al. 1997].

It has been suggested that true creative design is impossible for humans to achieve since human creativity must stem from the experiences and knowledge of the designer and is therefore systematic to a degree [Hashelmian et. al. 1997]. However, this fallibility of the human may be overcome to some degree in the following ways: increased human objectivity; by the use of "creative phases" where the individual tries to deliberately think against the design goals or by the use of larger design teams [Nezel 1995]. In practice the third option is the more common approach. However, increased use of design teams is by no means a guarantee of creative thinking, since there is often conflict in viewpoint, knowledge and confidence within the team [Ullman et. al. 1995]. The effects of these factors may be augmented if an individual has a desire for their design to be chosen, this coupled with the fact that seldom does a single design perform well in all criteria may reduce the degree of objectivity [Mill 1993].

By its very essence, systematic design overcomes the problem of designer objectivity however in doing so creativity is lost [Evatt 1995]. The application of creative or systematic design alone will not provide optimal results, since the analysis and implementation of creative solutions requires some logical and structured thinking [Haapasalo et. al. 1997]. It is therefore desirable that the designer should exhibit both systematic and creative design ability and should also be able to continuously change from skill to skill during design [Dixon 1966]
1.4 Computers and Design

The systematic nature of the computer has generally limited its application to the detailing stage [French 1985]. Here the power and accuracy of the microchip is used to exhaustively evaluate the performance of slight modifications to existing designs in an attempt to improve performance. This “brute force” approach was evident from the very first computing machine, developed to evaluate the ranges and elevations for military weapons. There have been attempts to use computers during creative design, but in mechanising creativity one moves towards a systematic process and innovation is lost [Evatt 1995]. With the advent of more powerful systems, computers are now being used in increasingly more complex environments. Decision support systems use modelling and virtual reality techniques to allow the design team to animate and interact with prospective designs [Richir et. al. 1997]. Multiple design alternatives may be quickly analysed and compared, thereby giving greater assurance that the final alternative is the optimal solution. However, even in these comparatively advanced systems, the main source of creativity is still the human user. Recent work has redressed the computer creativity problem by introducing Artificial Intelligence (AI) techniques into the design process. Gero [Gero 1994] suggests that AI can be used to provide a scheme to model human design and also to allow for the development of tools for human designers. Of the available AI techniques, Fuzzy logic and Expert systems have been used to model the human design and evaluation process [Andreichiov 1997 & Roy 1996], whilst Evolutionary Computing has been used to support engineering design [Gen 1997]. It is Evolutionary Computing that is discussed in the following sections.

1.5 Evolutionary Computation

In 1859, 27 years after his cruise on HMS Beagle and shortly after his death, Charles Darwin’s work on the origin of the species was published [Darwin 1859]. Darwin noticed that environmental changes were responsible for producing biological alterations brought
about by sexual reproduction. He further noted that although reproduction generally made living things depart from their ancestry, it does not guarantee that such deviations will be useful. Darwin proposed that there must be some principle that directed such evolution into profitable channels. After exhaustive research he concluded that the fertility of nature coupled with limited resources caused a struggle for existence. Within such a struggle, any individual that inherits a favourable parental trait would stand a better chance of survival and passing on the trait into the next generation. Darwin called this force natural selection [Miller 1986].

One factor Darwin did not answer was the biological mechanism of passing characteristics between parents. Within biological systems, the exchange or recombination occurs at the DNA level, and is caused by the displacement of a single (host) DNA strand by a further (donor) strand. This in turn causes the linking of two duplexes by crossed single strands, if the single strand pairs are broken they may reform duplexes that have recombined with each other. Hence, characteristics from one DNA strand pass to the other strand. Mutation is a further factor that may change the structure of DNA and involves the switching of the nucleotides (the basic DNA component) within the DNA duplex. Often mutation produces changes that are detrimental. However in the rare circumstances where such changes are beneficial, natural selection forces their incorporation into the evolutionary process.

The field of Evolutionary Computation seeks to mimic the ideas suggested by Darwin. This is achieved by four basic elements: a population of individuals; a notion of "fitness"; a birth / death cycle biased by fitness and a notion of "heritability" [DeJong 1999]. The evolutionary computing paradigm is given in figure 1.2.
Randomly or otherwise generate initial population
Until some end criterion is met
Select individuals based upon fitness from the population
Produce offspring
Select individuals to die
Introduce offspring into population

Figure 1.2: Evolutionary computing paradigm

The initial population is comprised of a number of candidate solutions to the problem to be solved. The fitness of an individual is defined by a model of the problem, thereby allowing for discrimination between the “better” and “poorer” solutions within the population. The fitter members are then selected to take part into the reproduction phase, these individuals are referred to as parents. The reproduction stage involves the transformation of the parents to form a given number of modified solutions, called children. A number of individuals are then selected from the overall population to be replaced by the child solutions. The new population, consisting of a mixture of parental and child solutions is referred to as the next generation (in some cases the next generation consists entirely of child solutions). The process then returns to the selection phase, where the next generation of parental solutions is chosen. This process continues until some stopping criterion is met.

There are four branches of evolutionary computation, Evolution Strategies, Evolutionary Programming, Genetic Programming and Genetic Algorithms. Evolution Strategies (ES) were developed by Rechenberg [Rechenberg 1965] and Schwefel [Schwefel 1965]. ES focus upon real valued parameter optimisation where each solution is a vector of real-valued parameters. The first ES involved the generation of a single child from a single parent. In the so-called (1+1) ES reproduction is asexual, a single child is generated by adding normally distributed random numbers to the lone parent. More sophisticated ES strategies include, (µ+λ) and (µ,λ). In the (µ+λ) ES, µ parents produce λ children (λ>µ),
from these $\mu+\lambda$ individuals the fittest $\mu$ become the parents of the next generation (the parents are randomly paired). The $(\mu,\lambda)$ ES differs slightly from the $(\mu+\lambda)$ approach since the $\mu$ parents are discarded immediately after producing $\lambda$ children, hence the life span of a single individual is limited to one generation. The parents for the next generation are the fittest $\mu$ children. In both cases mutation is the same as that used in the $(1+1)$ ES. During recombination, each component of the child vector is generated by either taking one of the corresponding parental vector values (discrete recombination) or by calculating the average of both (intermediate recombination).

Evolutionary Programming (EP) was developed in the 1960s by Fogel [Fogel 1966]. EP differs from ES since asexual reproduction is achieved through mutation alone. During a generation, $m$ parents are randomly chosen from the population of $m$ individuals to produce an equal number of children, a tournament is then held between the child and the parent with the individual of higher fitness survives into the next generation.

The final Evolutionary Computing technique discussed here is Genetic Programming (GP). GP was developed by Koza in the late 1980s to evolve computer programs by means of natural selection [Koza 1992]. Each chromosome within an evolving GP population is a computer program that attempts to solve the problem at hand. The programs consist of a number of traditional functions, such as arithmetic operators ($+, -, \times, +, \text{etc}$), Boolean (AND, OR, NOT), logical (IF, THEN, ELSE) and so on. One of the more popular uses for GP is in the field of regression. Consider an unknown function of the form, $z = f(x,y)$. GP finds $f(x,y)$ by firstly randomly generating a population of possible solutions. An example structure represented by the $i$th population member is given in figure 1.3.
Figure 1.3: The structure of a GP program (the nodes are represented by circles)

Reading in a bottom up manner, the function represented by the tree of figure 1.3 is,

\[
z = \left(\frac{2-x}{y}\right) + 5
\] ...

The fitness of this solution is given by a measure of the similarity between this and the actual function, more usually this is the sum error between the GP and actual function taken over a number of test cases. Reproduction involves the production of children by swapping parts or branches from two selected parents. The points at which branches are switched are chosen randomly for both parents. Mutation is achieved by introducing a random change to a single node.

1.5.1 Genetic Algorithms

Genetic Algorithms (GAs) form arguably the most active area of EC research. GAs are search algorithms that are widely used to solve optimisation problems. Given some function, \( y = f(x) \), where \( x \) and \( y \) are input and output vectors, the GA will seek the combination of input values that produces the optimal (maximal or minimal) output vectors.
The pioneer of the Genetic Algorithm was John Holland, in his book “Adaption in Natural and Artificial Systems” he defines the basic principles of the GA. In his doctoral thesis DeJong [DeJong 1975], who was one of Holland’s students, investigates the effects of population size, crossover and mutation upon the performance of a GA, it is here where the first standardised set of test functions and performance measures relating to GAs were proposed. More recently, Goldberg [Goldberg 1989] published his book “Genetic Algorithms in Search Optimisation and Machine Learning, which is regarded by many as the seminal book within the GA field. The 10 years since the publication of Goldberg’s book has seen a massive increase in the degree of research relating to GAs. Today, the application of GAs to practical real world domains continues unbounded.

The canonical GA paradigm is shown in figure 1.4. As with the other EC approaches, it is assumed that the problem may be represented by a vector of input parameters (the hypercube represented by the envelope of the input vector is called the search, design or input space). These input parameters are referred to as genes and join together to form a string of values that collectively constitute an individual chromosome. Often, the chromosome is represented by a binary string, this encoded information is called the genotype and must be decoded into the set of true input values (phenotype) before the corresponding fitness can be calculated (section 1.5.1.1). The encoded genotype therefore represents the DNA of the individual. The population, typically initialised by randomly generating either a 0 or 1 at every point within each chromosome, is then decoded and the fitness of each chromosome calculated by the objective function. Prospective parents are then selected in general proportion to their fitness (section 1.5.1.2), this stage is similar to natural selection as described by Darwin. The reproduction phase or crossover involves the swapping of binary information between selected parents and represents the interchange of DNA fragments. Finally mutation, where individual chromosome bits are flipped is analogous to the switching of nucleotides within the DNA. At this point, the
population is again decoded and the individual chromosome fitnesses are calculated. If a
stopping criterion (usually based upon either the number of generations run or the fitness
of the best chromosome) is met search ends, otherwise the process loops back to the
selection phase and a new generation begins.

The population therefore evolves over successive generations, characterised by a gradual
increase in both the best and average fitness within the population. This process is referred
to as the convergence of search. However, it is by no means desirable for search to
converge as quickly as possible. Efficient search is characterised by the correct balance
between exploration and exploitation. Exploration is the search of new and unknown
regions within the search space whilst exploitation is search within identified regions to find solutions of greater optimality. These two characteristics are dichotomous, and GA search must find the best compromise between them both.

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allele</td>
<td>The value of a gene. In a binary chromosome, each gene may have an allele of 0 or 1.</td>
</tr>
<tr>
<td>Gene</td>
<td>A subset of a chromosome that usually encodes the value of a single parameter.</td>
</tr>
<tr>
<td>Chromosome</td>
<td>A string of genes.</td>
</tr>
<tr>
<td>Genotype</td>
<td>The encoded parameter set</td>
</tr>
<tr>
<td>Phenotype</td>
<td>The decoded parameter set</td>
</tr>
<tr>
<td>Gene pool</td>
<td>The collection of genes in the population</td>
</tr>
<tr>
<td>Generation</td>
<td>The process of creating a population of children from a population of parental chromosomes</td>
</tr>
<tr>
<td>Search space / design space</td>
<td>The space described by the bounds of the n optimised variables</td>
</tr>
<tr>
<td>Convergence</td>
<td>The evolution of search characterised by an increase in the average and best fitness within the population</td>
</tr>
<tr>
<td>Exploration</td>
<td>The search of new and unknown regions within the search space</td>
</tr>
<tr>
<td>Exploitation</td>
<td>Search within explored regions with the aim to find solutions of greater optimality</td>
</tr>
</tbody>
</table>

**Table 1.1: Definition of terms**

To understand how a GA works, one must firstly understand what is meant by the term schemata. The Hitchhiker’s Guide to Evolutionary Computation defines Schemata as “a pattern of gene values within a chromosome” [Heitkotter et. al. 1994]. More simply, schemata are used to define any bit pattern within a chromosome. The schemata alphabet consists of three symbols 0, 1 and *. The asterisk represents either a 0 or a 1 and is often referred to as a “don’t care” symbol. For example, the following binary strings 1000, 1010, 1100 and 1110 are all examples of the schemata 1**0. A schema is further characterised by its order and defining length. Order is the number of fixed (1 or 0) bit positions within the schema, the order of 1**0 is two. The defining length is the difference between the first and last binary points, the defining length of 1**0 is three (4-1). In 1989, Goldberg’s schema theorem [Goldberg 1989] showed that low order schema of short defining length and above average fitness receive exponentially increasing trials during search. Goldberg calls such schemata building blocks and suggests that a GA works by the
continual juxtaposition of building blocks that generate solutions of increasingly higher fitness. It is interesting to note the parallel between this and the notion suggested by French [French 1985], that creative design is often a series of small incremental steps rather than any single leap of inspiration.

The following sections give a review of the GA mechanisms and strategies that are relevant to this thesis. It should be noted that no attempt is made to provide a comprehensive analysis. The more interested reader should refer the following texts for a more wide reaching review, [Goldberg 1989, Michalewicz 1992, Gen et al. 1997 & Man et al. 1999].

1.5.1.1 Encoding
The are two types of encoding used to represent parameters within the chromosome, binary encoding and real parameter. Real parameter or floating point representation encodes each chromosome as a vector of floating point numbers that is of the same size as the solution vector (figure 1.5).

\[
chromosome_j = [708.45, 0.000234, ... , 1.2 \times 10^7]
\]

\[
\text{variable}_1 \quad \text{variable}_2 \quad \text{variable}_n
\]

**Figure 1.5:** Floating point representation

The obvious advantage of floating point representation is that the value of each variable does not need to be encoded in any way, as a result such phenotypic representation allows for any degree of precision within the search space. However, the crossover and mutation techniques used within real-parameter encoding are not consistent with application to binary encoded chromosomes. Although there is some empirical evidence to suggest the superiority of floating point over binary representation [Janikow et al. 1991] such findings
are by no means conclusive [Man et. al.1999]. As a result, floating point representation is not used in this thesis and is not discussed further.

With binary encoding, each input variable is coded into a binary segment, which collectively constitutes the chromosome (figure 1.6). Before the fitness of a solution may be determined the binary chromosome must be decoded into the $n$ input variables, this is achieved by executing a mapping from the representation (genotypic) space to the input or design (phenotypic) space.

\[
\text{chromosome}_j = 001010100\ 01101110\ldots\ldots\ldots\ldots11010101
\]

\[
\text{variable}_1\ \text{variable}_2\ \text{variable}_n
\]

**Figure 1.6:** Binary representation

The encoding resolution of the $i$th variable is given by,

\[
\text{resolution}_i = \frac{\max_i - \min_i}{2^{N_i} - 1}
\]

The decoded (true) value is given by,

\[
\text{true}_i = (\text{binary}_i \times \text{resolution}) + \min_i
\]

where,

- $\max_i, \min_i =$ maximum and minimum value of the $i$th variable
- $\text{true}_i =$ true value of the $i$th variable
- $\text{binary}_i =$ decoded binary integer of the $i$th variable
- $N_i =$ number of bits used to encode the $i$th variable

Consider the $j$th chromosome of a binary encoded population representing two variables, $x$ and $y$, of the range,
\[-5.0 \leq x \leq 5.0 \]
\[-5.0 \leq y \leq 5.0 \]

and,

\[
\text{chromosome}_j = 011010\,100110
\]

NB (the bits relating to variable x are highlighted in dark grey, the bits relating the variable y are highlighted in light grey).

If six bits are used to encode each variable \((N = 6)\), using equation 1-2, the corresponding resolution of both variables is,

\[
\text{resolution} = \frac{5 - (-5)}{2^6 - 1} = 0.158730
\]

The binary segments relating to each variable and their corresponding decoded binary integer values (\(\text{binary}_x\) and \(\text{binary}_y\)) are,

\[
x = 011010 \quad \text{binary}_x = 2^4 + 2^3 + 2^1 = 26
\]
\[
y = 100110 \quad \text{binary}_y = 2^5 + 2^2 + 2^1 = 38
\]

Applying equation 1-3, the true decoded values for \(x\) and \(y\) are found to be,

\[
\text{true}_x = (26 \times 0.158730) + (-5) = 0.87302
\]
\[
\text{true}_y = (38 \times 0.158730) + (-5) = 1.03174
\]

There exists a special binary encoding strategy designed specifically to handle a mixture of discrete and continuous variables. Structured encoding [Dasgupta 1991], uses chromosomes that contain a number of redundant segments, these relate to differing solution configurations dictated by the discrete variables. During search the discrete variables "switch on" and make active specific segments within the chromosome. Consider a simple structured chromosome consisting of four variables one discrete and
three continuous. The discrete variable \((d)\) corresponds to three different solution configurations donated by, \(d_1, d_2\) and \(d_3\). The continuous variables \(x, y\) and \(z\) each have differing ranges depending upon the value of \(d\), these being denoted by: \(x_1, x_2\) and \(x_3\); \(y_1, y_2\) and \(y_3\) and \(z_1, z_2\) and \(z_3\). If the discrete variable corresponds to \(d_2\) only \(x_2\), \(y_2\) and \(z_2\) are active and produce a fitness value for the chromosome (figure 1.7). It should be noted that when applied to a structured chromosome, crossover and mutation remain unaffected, and may still modify the redundant segments.

![Figure 1.7: The structured chromosome (the highlighted variables indicate active areas within the chromosome, \(d, x, y, \) and \(z\), represent binary segments relating to each variable)](image)

### 1.5.1.2 Selection Strategies

The most widely employed selection methods belong to a group called fitness proportional selection (FPS) [Hancock 1994]. FPS selects chromosomes in proportion to their fitness. Roulette wheel selection (RWS) or stochastic sampling with replacement is the most common example of FPS. Figure 1.8 shows the RWS methodology when applied to a population of 10 chromosomes.

The proportion \((P_j)\) the \(j\)th chromosome takes of the roulette wheel is given by,

\[
P_j = \frac{f_j}{\sum_{i=1}^{n} f_i}
\]

...1-4

where,

\[
\begin{align*}
  n & = \text{number of chromosome (population size)} \\
  f_i & = \text{fitness of } i\text{th chromosome}
\end{align*}
\]
The pointer is independently spun \( n_{\text{cross}} \) times to select \( n_{\text{cross}} \) chromosomes for reproduction (\( n_{\text{cross}} \) is the number of individuals that take part in crossover and is typically 60% of the population size [DeJong 1975]). Hence the higher the fitness of an individual the more likely it is of being selected. Deterministic sampling differs slightly since the expected number of samples for each individual \( (e_j) \) is calculated to the nearest integer,

\[
e_j = \lfloor P_j \times n_{\text{cross}} \rfloor
\]

...1-5

The population is then sorted in descending value of \( e_j \) and individuals are selected \( e_j \) times for reproduction. If further chromosomes are needed they are taken from the top of the sorted list. Stochastic tournament selects successive parents in the same manner as RWS, however only the child with the higher fitness is inserted into the population, this process being repeated \( n_{\text{cross}}-1 \) times.

A significant drawback of the majority of FPS methods is that it is possible that a highly fit chromosome (denoted by \( P_2 \) in figure 1.8) may be selected (or sampled) a disproportionate number of times. Baker [Baker 1987] calls this phenomenon sampling bias and argues that it may lead to the premature convergence of search due to the domination of the gene pool...
by a few highly fit individuals. Baker proposed a new FPS selection scheme called stochastic universal sampling (SUS) which has zero sampling bias. SUS uses the same roulette wheel as RWS, however \(n_{cross}\) pointers are used to select all the required chromosomes simultaneously (figure 1.9). The equally spaced pointers are \(\theta_{pointer}\) degrees apart where,

\[
\theta_{pointer} = \frac{360}{n_{cross}}
\]

...1-6

From figure 1.9 it can be seen that bias is eliminated by a much reduced likelihood that chromosome two will be selected a disproportionate number of times.

1.5.1.3 Crossover Operators

Of the numerous crossover mechanisms developed, the most widely used are one and two-point crossover. As its name suggests, one point crossover is defined by a single, randomly generated point along the length of the chromosome (figure 1.10). The binary bits of each parental chromosome (\(P_1\) and \(P_2\)) immediately after the “cut point” are then interchanged to form the children (\(C_1\) and \(C_2\)), forcing the interchange of genetic material.

![Figure 1.10: One point crossover](image)

![Figure 1.11: Failure of one point crossover](image)

In certain instances, there is a problem with single point crossover, consider the two parents of figure 1.11. Parent 1 contains two schema, whilst parent two contains a single schemata. If single point crossover is applied, it is impossible that any subsequent child will contain all three schemata within its genotype [Michalewicz 1996]. As a result of this,
multi-point crossover, of which double point is an example, was developed. Double point crossover uses two randomly chosen cut points (figure 1.12). The exchange of binary information then occurs between the two cut points, as a result the reduction in schema sampling with single point crossover is far less evident (figure 1.13).

![Figure 1.12: Two point crossover](image)

Crossover points:

- **P1:** 01000011110
- **P2:** 01000110110

**C1:** 01000011110
**C2:** 01000110110

![Figure 1.13: Success of two point crossover](image)

Crossover point:

- **P1:** 01000011110
- **P2:** 01000110110

**C1:** 01000011110
**C2:** 01000110110

Uniform crossover is a more recent approach and has been empirically shown to be more effective at recombining schemata then either one or two-point crossover [Syswerda 1989]. Uniform crossover works by firstly generating a binary template equal to the length of both parents, each bit within **P1** and **P2** is then checked against the template. If the corresponding bit value of the template is one the parents exchange genes, if the value is zero, the parents retain their genes (figure 1.14). Eshelman implements a modification of Uniform crossover in the HUX operator used within the CHC algorithm [Eshelman 1991]. In this instance a mask is generated that swaps exactly half of the parental bits.

![Figure 1.14: Uniform crossover](image)

Crossover points:

- **P1:** 0110001001110
- **P2:** 010000110110

**RS P1:** 010000110110
**RS P2:** 0110001001110

**C1:** 010000110110
**C2:** 0110001001110

![Figure 1.15: Double point reduced surrogate crossover](image)

In this instance a mask is generated that swaps exactly half of the parental bits.
One drawback of all of the crossover techniques discussed so far is a reduction in the ability to sample parental schema as the population converges and becomes more similar. If has been suggested that this factor contributes to premature convergence as diversity is lost by “needlessly ineffective” crossover operations [Booker 1987]. An example of this is when selected parents are different but the crossover points fall so as to exchange identical genes, hence the children will be identical to the parents. Booker proposed a crossover technique called double point reduced surrogate crossover, which constrains crossover to produce variations where possible. This is achieved by firstly extracting the reduced surrogates of each parent, \( RS \ P_1 \) and \( RS \ P_2 \) of figure 1.15. Double point crossover is then performed upon the reduced surrogates to produce \( RS \ P_1^* \) and \( RS \ P_2^* \), these are then reinserted into the parents to produce the child chromosomes.

### 1.5.1.4 Mutation Operator

Mutation is a relatively simple process and involves the random flipping of a single bit to introduce variation into a chromosome (figure 1.16). In doing this, mutation can reintroduce schema that have either been lost from the gene pool by the action of selection or crossover, or that did not exist at the point of initialisation. The number of mutations made per generation is governed by the mutation probability, \( p(mutation) \), this being the probability that a randomly chosen bit will be mutated. The mutation probability is typically in the range 0.001 [DeJong 1975] – 0.01 [Grefenstette 1986].

![Before and after mutation](image)

**Figure 1.16:** The action of mutation
1.6 Thesis Overview

This thesis has commenced with a brief overview of engineering design, outlining the relationship between systematic and creative design and how computers have traditionally been used to support such design processes. The field of Evolutionary Computing has been reviewed with particular emphasis placed upon the philosophy and specific mechanisms of Genetic Algorithms. Chapter two introduces Cluster Oriented Genetic Algorithms (COGAs) and more specifically variable mutation COGA (vmCOGA), which is treated as the archetypal COGA algorithm throughout the thesis. Differences between COGA and multi-modal algorithms are also highlighted. The chapter continues with an examination of the individual COGA components, including the diverse search engine and the adaptive filter. The relationship between the filtering vector and the final clustering set is demonstrated by application to a simple 2d-test function. The role COGA plays in the conceptual design stage is discussed in chapter three, by applying it to a number of commonly encountered design scenarios described by two real-world multi-dimensional models. The integration of COGA with a graphical user interface and the proposed Interactive Evolutionary Design Station (IEDS) is also examined. In chapter four the viability of replacing the initial COGA search engine with a number of "off the shelf" diverse search and multi-modal algorithms is considered. The prospective algorithms are tested upon two multi-modal 2d-test functions and a novel 6d-test function. Chapter five investigates two innovative COGA search algorithms, HiCOGA and SsCOGA, designed to increase exploration and sampling of isolated peaks within the search domain. The performance of these algorithms is compared with vmCOGA and it is shown that the degree of exploration may be altered by manually modifying the search parameters of either algorithm. One of the main drawbacks of HiCOGA and SsCOGA is identified as being the need to select the search parameters before search commences. Consequently, chapter six focuses upon the development of a control mechanism that automatically adapts the parameters during search by using a feedback measure based upon population
convergence. The performance of the adaptive algorithms is compared against vmCOGA on six test functions of increasing dimensionality. The performance of the adaptive filter is investigated in chapter seven. A more efficient filtering technique called Dynamic adaptive filtering (DAF) is proposed and integrated with vmCOGA and the more successful adaptive algorithms of chapter six. Empirical evidence suggests that the DAF significantly improves the robustness of COGA. Chapter eight addresses the final component of the COGA mechanism, knowledge discovery within the final clustering set. A framework is suggested for how the results of a COGA run may be most efficiently presented without overburdening the designer with information overload. Chapter nine reviews and concludes the thesis. Future areas of research and possible development directions are discussed at this point.
Chapter 2

Cluster Oriented Genetic Algorithms (COGAs)

2.1 Introduction

Cluster Oriented Genetic Algorithms (COGAs) have been developed by Parmee [Parmee 1996a & 1996b] to assist the designer or design team\(^1\) during the conceptual design stage by supporting the initial exploration of complex design spaces. This is achieved by a rapid decomposition of the search space into succinct regions of high performance (HP). Extracted qualitative and quantitative information, personal expertise and more specific "in-house" knowledge such as manufacturing and material costs allow the designer to assess the relative performance of each decomposed region. Successive COGA runs may then investigate specific areas of the search space where novel design directions may be found. Throughout this iterative process the designer continuously gains general knowledge relating to the overall design domain, such as design variable sensitivity, design variable / objective interaction, constraint violation, multi-objective criteria etc., and more specific information relating to the regions of high performance, location, appearance etc.

The conceptual design environment is usually characterised by poor domain knowledge, uncertainty relating to objectives and fuzziness relating to design variable / output interaction [Parmee 1997]. Consequently, conceptual design models are coarse representations of the system under investigation. The validity of single solutions from such design models should therefore be treated with caution, since they may be symptomatic of local high performance or at worst erroneous [Parmee 1997]. This coupled

\(^1\) Designer and design team are interchangeable terms used to describe the user or users of COGA.
with the disproportionate computation effort (in terms of local search) required locating optimal solutions renders the identification of single optima infeasible within such an environment. Regional decomposition overcomes this uncertainty and also extracts information throughout the search reducing the need for further localised investigation.

The requirements of a decomposition algorithm such as COGA are as considerable as they are demanding. Firstly, runtimes should be sufficiently small to allow for operation to be approaching “real-time”. The number of solutions populating high performance regions should be as high as possible, whilst still maintaining a crisp definition of the decomposed region. To reduce computational expense, the number of function calls should be as low as possible. Since COGA is used in an iterative manner, it is critical that robustness allows for use within any perceived design domain. Finally, the results generated by COGA should be presented to the designer in a manner that maximises presented information whilst minimising the chances of information overload.

The generic mechanism of a COGA run is shown in figure 2.1. Firstly the search space is specified by setting the maximum and minimum value of each of the design variables. Secondly the search objective is set, this relates to the optimisation of any one of the design outputs or may involve multi-objective or constraint violation optimisation. Finally the generation, mutation and filtering vectors are set and search commences. Upon completion, the high performance regions identified during search are presented to the designer along with design information relating to each region.

The overall objective of COGAs is therefore the identification of high performance regions and the achievement of sufficient regional set-cover (in terms of number of solutions) to allow significant qualitative and quantitative information to be extracted. Using this information and information from successive runs, design expertise and in house
knowledge; the search parameters, objectives, constraints etc. may be modified, to investigate the feasibility of novel search areas or the further development of established search directions.

Figure 2.1: The COGA mechanism

In terms of evolutionary based techniques, the closest relative to COGAs are Multi-Modal Optimisation (MMO) GAs. Upon initial inspection, there appears to be little difference between both techniques, however this is not the case. This chapter begins with a comparison of MMO and COGA, highlighting the main differences in both techniques and discussing the problems encountered when attempting to apply MMO algorithms to real world design models.
2.1.1 COGA and Multi-Modal Optimisation

Multi-Modal Optimisation (MMO) concerns the location of multiple solutions of optimal or near optimal fitness contained within a given function or model. There are several MMO techniques, each possessing particular characteristics, advantages and disadvantages, however one factor they all share is an inspiration taken from the natural world.

2.1.1.1 Crowding

Crowding is based upon the notion that in nature dissimilar individuals are more likely to occupy independent niches (regions of high performance), and will therefore be less likely to reproduce. Conversely, similar individuals will occupy the same niches and will be far more likely to reproduce. When the number of individuals populating a niche becomes too great, the older and weaker members of the niche are forced out by the younger and stronger members. This results in a continuous recycling of individuals (species) about a peak.

Crowding GAs mimic this by forcing local rather than global competition, this is achieved by replacing population individuals with phenotypically or genotypically similar individuals. The first documented Crowding GA was Preselection [Cavicchio 1970], here a child chromosome (selected on a fitness proportionate basis) replaces the inferior parent. DeJong Crowding [DeJong 1975] is a development of Cavicchio's Crowding where for every n offspring CF (Crowding Factor) population members are randomly selected. The member of CF with the closest genotypic similarity (Hamming distance) to the offspring competes with it on a fitness basis for survival. DeJong noted a high incidence of replacement errors (one individual replacing another individual from an independent niche), leading to the inability to maintain stable niches within the population. This coupled with both the need to select a value for CF and the insensitivity of genotypic
similarity, since each bit in the chromosome has equal significance, limits its success. Deterministic Crowding (DC) [Mahfoud 1992] significantly reduces replacement error by increasing the probability of both parent and child belonging to the same niche. Firstly the population is randomly paired, reproduction and mutation then take place to produce two children. The two children compete with the most phenotypically (Euclidean distance) similar parent for survival. DC has many advantages over the aforementioned Crowding algorithms, the reduction of replacement errors, the replacement of genotypic with phenotypic distance metric and the elimination of parameter calibration before running. However, DC is unable to maintain stable niches within the population. The possibility of eliminating niches during search is therefore high. More recently, Multi-Niche Crowding [Cedeno et. al. 1995] has developed the general Crowding methodology, however this technique requires the tuning of three crowding factors, placing further emphasis upon the user.

2.1.1.2 Sharing

Within nature, a niche has a finite amount of resources to give to the individuals that occupy it. As the number of individuals increase, the resources must be shared. There may come a time where an individual will receive a greater resource share if it migrates to another, less densely populated niche. Consequently, a near equilibrium state emerges where each peak supports a given number of individuals dependent upon the nature and location of the niche. Individuals who do not receive sufficient resource share from a given niche must locate further niches in order to reproduce into the following generations.

The most common Sharing algorithm is Fitness Sharing [Goldberg et. al. 1987]. Fitness Sharing reduces the fitness of a chromosome by an amount dependent upon the number of similar individuals within the population. An individual is considered to be similar if its distance metric $d_i$ is less than a threshold value, $\sigma_{\text{share}}$. The number of individuals within
$\sigma_{\text{share}}$ of a given solution and also a constant $\alpha$, determine by what factor the fitness of an individual is reduced. As with the majority of the Crowding schemes, the main drawback of Fitness Sharing is the need to select values for $\sigma_{\text{share}}$ and $\alpha$ ($\alpha$ is a constant that determines the amount by which the fitness is shared). Much research effort has been targeted at solving this problem. Deb et. al. [Deb et. al. 1989] give formulae for determining $\sigma_{\text{share}}$ ($\alpha$ being set to 1), however the number of peaks within the search space is required. Yin and Germay [Yin et. al. 1993] reduce the need for apriori by clustering the population, the fitness of an individual being reduced in proportion to the number of individuals in the cluster to which it belongs.

2.1.1.3 Restricted Tournament Selection

As with the Crowding techniques of section 2.1.1.1, Restricted Tournament Selection (RTS) seeks to replace similar members of the population [Harik 1995]. However, unlike the generational approach of Crowding, RTS uses steady state populations, this means a child may be selected for crossover immediately after it has been created. Two parents are selected at random for reproduction, the corresponding child then competes for survival with the closest member of a subset of size $w$, selected randomly from the population. Unsurprisingly, the main drawback of RTS is the need to set the value of $w$, if the value is too small the probability of replacement errors is high. Adaptive RTS (ARTS) [Roy et. al. 1995] overcomes the need to select such a value. The population is firstly sorted into clusters using Shared Nearest Neighbour Clustering [Jarvis et. al. 1973] and the relevant child then competes for survival with the nearest individual from the closest cluster, thereby significantly reducing the probability of replacement errors. However, this technique is not without disadvantages, the most significant of which being the need to select two parameters ($K$, the size of the nearest neighbourhood list and $KT$, the number of compared elements within the nearest neighbourhood list) for the clustering algorithm.
2.1.1.4 Sequential Niching

Breaking away from the Crowding and Sharing philosophies, Sequential Niching [Beasley et. al. 1993] searches through the design space until a peak is identified. The local fitness of the peak is then reduced and search is reinitialised. The utility of this approach is again reduced by the need to set values to achieve the depression of the fitness landscape, furthermore, the possible loss of schema located within the depressed regions may reduce the diversity of search. These disadvantages coupled with the larger runtimes caused by the sequential nature of the algorithm has limited the further development of this technique.

2.1.1.5 Parallel Methods

The most recent field of MMO research involves the simultaneous evolution of multiple populations. The ECOlogical GA (ECOGA) [Davidor 1991 & Davidor et. al. 1993] forces reproduction amongst similar individuals, by placing each chromosome on a two-dimensional toroidal grid (one chromosome per grid element). The first parent \((P_1)\) is selected randomly from the grid and reproduced with parent two \((P_2)\), which is selected from eight grid members adjacent to \(P_1\) (the eight grid members along with \(P_1\) form a temporary subpopulation). For each child a tournament for survival is held between itself and a randomly chosen chromosome from the subpopulation. At initialisation the population is randomly placed upon the grid, however after a number of generations, groups of similar solutions start to emerge upon the surface of the grid. In this way children are more likely to stay in the vicinity of their parents, the overall effect of the ECOGA is rapid local convergence whilst maintaining global diversity and speciation. However, the overlapping nature of the subpopulations will result in the eventual domination of the grid by a single niche. Three grid sizes are investigated \((32x32, 45x45\) and \(71x71)\) suggesting that the population size is by no means arbitrary. Further parallel GAs have been developed by Elo [Elo 1994], in this example search begins within a single
population, that is divided into two subpopulations when a new niche has been discovered. Search continues and sub population division occurs repeatedly as new niches are discovered. The overall goal of this technique is to locate $n$ niches each contained within $n$ subpopulations.

With the exception of DC and the ECOGA, one of the continuously reoccurring drawbacks of the MMO algorithms discussed in this section is the need for apriori search space knowledge in order to set algorithm parameters. The incorrect setting of such parameters may seriously degrade performance. This problem is magnified when higher dimensionality models are investigated since visualisation of the search space becomes problematic and the correct calibration of algorithm parameters becomes progressively more critical. A further disadvantage of many MMO approaches is the inability to maintain stable regions of high performance within the population. This coupled with the relatively small search space cover caused by finite population size makes it highly probable that useful information relating to the niches may be lost during search.

COGA eliminates these disadvantages by utilising an adaptive filter. This continuously adapts to the design domain by accurately modelling the fitness distribution throughout search. Furthermore, the dynamic extraction and storage of high performance solutions throughout search renders obsolete the need to maintain stable niches within the population.

2.2. Generating COGA Solutions

Section 2.1 introduced the COGA philosophy and discussed how COGA is used within the design environment. The following sections expand the “generation solutions” block of figure 2.1 by discussing in more detail the most important aspects of COGA: the generation of solutions by the diverse search algorithm; the extraction of solutions by the
Adaptive Filter and the storage of solutions in a database referred to as the Final Clustering Set (FCS).

2.2.1 The Diverse Search Engine - Variable Mutation

The Diverse Search Engine explores the search space and locates high performance regions contained within it. During this search it is important to maintain diversity within the gene pool and to achieve and maintain the correct balance between exploration and exploitation [Mauldin 1984, Goldberg 1989, Booker 1987 & Tsutsui 1997]. Exploration (seeking new regions of high performance) is critical during the early stages of search in order to sample the entire search space thereby increasing the possibility of locating isolated regions of high performance. As search progresses the importance of exploration is gradually superseded by exploitation (searching within identified high performance regions), this increases the number of solutions located within high performance regions and adds to the database of solutions contained FCS.

The initial diverse search engine used within COGA utilised a variable mutation strategy [Parmee 1996a & 1996b]. Variable mutation COGA (vmCOGA) search is split into a number of search stages, each consisting of a number of generations (figure 2.2). During the initial search stage, high mutation ensures an efficient sampling of the whole search space. After the completion of each subsequent stage the mutation probability is reduced to promote the convergence of search and the formation of clusters of solutions upon high performance regions of the search space. Two vectors define search, the generational vector \( (g) \) defines the end point (in generations) of each search stage (equation 2-1), the mutation vector \( (m) \) defines the mutation probability, \( p(mutation) \) used during each search stage (equation 2-2),
The search vectors, \( g = \{5, 10, 15, 20, 25\} \) and \( m = \{0.08, 0.06, 0.04, 0.02, 0.01\} \) therefore define a vmCOGA run of five search stages each consisting of five generations, the mutation probability during stage one to five is 0.08, 0.06, 0.04, 0.02 and 0.01 respectively.
To further promote diversity, Stochastic Universal Sampling (SUS) [Baker 1987] and Double Point Reduced Surrogate Crossover [Brooker 1989] are used during search [Parmee et. al. 1997a]. Unless otherwise stated, 12 bit per variable binary encoding, a selection probability of 0.6 and a population size of 100 (both suggested by DeJong [DeJong 1975]) are used as defaults throughout this thesis.

### 2.2.2 The Adaptive Filter (AF)

The diverse search engine explores the design space and generates all information relating to it. However, on its own this process is of little use unless the information can be extracted. This information extraction is achieved by the Adaptive Filter (AF), which forms the information gathering component of COGA.

The AF copies the fittest members of the population to the FCS. One may argue that this may be achieved by copying a solution if its fitness exceeds a certain fitness threshold or more simply copying the fittest percentage of the population to the FCS. However both of these techniques have considerable disadvantages. The first method requires the setting of a fitness boundary, which reintroduces the need for search space apriori such as the maximum or minimum fitness. In certain cases the second method may introduce low fitness solutions into the FCS, consider a hypothetical search situation within a heavily constrained search space. In this instance the population may consist of a number of very low fitness solutions (subject to a given penalty function) and a number of solutions with relatively high fitness. Further assume that the proportions of these solutions within the
population are 75% and 25% respectively and that top 40% of solutions are copied to the FCS. It can be seen that a number of the low performance solutions will enter into the FCS. Although this is not catastrophic it may cause misinterpretation when the contents of the FCS are presented to the designer and should therefore be avoided.

A solution to both of these problems is to construct a model that determines the fitness distribution of the current population, deductions may then be made as to which solutions pass the filtering process. Ideally a deterministic model would be used, however the lack of apriori coupled with fluctuations in the fitness distribution between runs makes this approach infeasible. Probabilistic models offer more utility over deterministic models since they estimate what proportion of the population lies at different fitness values, using such models it is a relatively simple process to determine the fitness distribution of the population and to set the corresponding filtering.

The most commonly used probabilistic modelling approach is that offered by Probability Density Functions (PDFs) of which the normal and standardised normal distribution are the most prevalent.

2.2.2.1 Probability Density Functions

PDFs are useful since they allow the modelling of any continuous variable (such as fitness) from any sufficient number of observations (sample) of that variable (individual chromosome fitness). A variable is said to be continuous if it may assume an infinite number of points between bounds. Consider a continuous variable $x$; there is a function $f(x)$ such that,

$$\int_{-\infty}^{\infty} f(x) \, dx = 1$$

$$f(x) \geq 0 \quad \forall x$$

...2-3
The function is referred to as the probability density function of $x$ [Hayter 1996] and may be used to predict the proportion of the sample that lies between certain variable bounds, $a$ and $b$ (figure 2.3),

$$P(a \leq X \leq b) = \int_a^b f(x) \, dx$$

Where $X$ is a random variable of the range $x_{\text{min}} \leq X \leq x_{\text{max}}$

PDFs therefore eliminate the need for apriori search space knowledge as the PDF may be calculated directly from the sample data i.e. the population fitnesses. Since it may be used across a wide variety of engineering design domains [Scheaffer et. al. 1995] the AF uses the normal distribution to model the fitness distribution of the evolving population.

![Figure 2.3: A Probability Density Function](image)

2.2.2.2 The Normal Distribution

The normal distribution (figure 2.4) is symmetric about the mean (zero skewness, see section 7.1.2.2) and is characterised by large numbers of points lying about the mean value and progressively fewer as one moves further away in either the positive or negative directions [Hayslett, 1981]. In practice the normal distribution is used extensively to
model variables such as IQ, examination pass rates, thermistor performance, milling machine performance, air pollution and so forth.

The standardised normal distribution is a special case of the normal distribution where the mean and standard deviation of the distribution are transformed to zero and one respectively (figure 2.5). Any sample of data \((x_1, x_2, \ldots, x_n)\) may be normalised by using the normalisation equation,

\[
    x_i^* = \frac{x_i - \mu}{\sigma}
\]

Where, \(\mu\) and \(\sigma\) are the mean and standard deviation of the sample, \(x_i\) and \(x_i^*\) are the true and normalised values of the \(i\)th sample member.

\[
    f(x) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad -\infty \leq x \leq \infty
\]

\[
    p(X \leq x) = \frac{1}{\sigma \sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{(t-\mu)^2}{2\sigma^2}} \, dt
\]

\[
    \text{mean} = \mu = \frac{\sum x_i}{n}
\]

\[
    \text{sd} = \sigma = \sqrt{\frac{\sum (x_i - \mu)^2}{n-1}}
\]

Figure 2.4: Normal distribution \((\mu = 0.75, \sigma = 0.25)\)

\(n = \) number of observations

\(X\) is a random variable of the range \(x_{\text{min}} \leq X \leq x_{\text{max}}\)

The main advantage of the standardised normal distribution is the sample data is always mapped to a corresponding set of values with \(\mu = 0\) and \(\sigma = 1\). This is useful since it eliminates the need to know maximum and minimum objective function fitness.
2.2.2.3 Explicit Filtering

Explicit filtering occurs after each search stage. Individual population fitnesses are normalised ensuring that the mean and standard deviation of the population fitness distribution is zero and one respectively. Each chromosome within the population is then checked against a predefined filtering threshold \( R_f \). If the normalised fitness of the chromosome is greater than \( R_f \), the solution is copied to the FCS (figure 2.6.). If the normalised fitness is less than \( R_f \) the does not enter the FCS.

As with the diverse search engine, the filtering thresholds applied during search are defined by a filtering vector, \( R_f \)

\[
R_f = \begin{bmatrix}
\text{filtering threshold used at the end of search stage 1 (} R_{f_1} \text{)} \\
\text{filtering threshold used at the end of search stage 2 (} R_{f_2} \text{)} \\
\text{filtering threshold used at the end of search stage 3 (} R_{f_3} \text{)} \\
\vdots \\
\text{filtering threshold used at the end of search stage n (} R_{f_n} \text{)}
\end{bmatrix}
\]

Hence, the filtering vector, \( R_f = \{1.5, 1.5, 1.5, 1.5, 1.5\} \), defines five explicit filtering stages each using a filtering threshold of 1.5.
2.2.2.4 Implicit Filtering

Explicit filtering results in a significant loss of information since solutions generated during intermediate generations (generations between filtering stages) are not copied to the FCS. To overcome this a further filtering mechanism is used. Implicit filtering occurs during all intermediate generations (after the completion of the first search stage) and copies a solution to the FCS if its unscaled fitness exceeds the fitness value associated with the previous filtering threshold [Parmee 1996a & 1996b]. Consider an arbitrary filtering threshold value $R_f$ (used to filter the population at the end of the $i$th search stage), and a population of chromosomes with mean fitness $\mu$ and standard deviation $\sigma$, taken at a point between filtering stage $i$ and $i+1$. The fitness associated with the filtering threshold value $(fit_{R_f})$, may be obtained by rearranging the normalisation equation (equation 2-5),
\[ \text{fit}_R = \sigma Rf + \mu \]

Consequently, a solution is copied to the FCS during any of the intermediate generations if its true fitness exceeds \( \text{fit}_R \).

The differences between explicit and implicit filtering are outlined in table 2.1.

<table>
<thead>
<tr>
<th>Explicit Filtering</th>
<th>Implicit Filtering</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Occurs at the end of each search stage.</td>
<td>• Occurs during all intermediate generations (generations between filtering stages) after the completion of the first search stage.</td>
</tr>
<tr>
<td>• The mean and SD population fitness are calculated, and the fitness of each chromosome is normalised.</td>
<td>• The mean and SD population fitness are calculated.</td>
</tr>
<tr>
<td>• A chromosome is copied to the FCS if its normalised fitness is greater than the corresponding ( Rf ) value within the filtering vector.</td>
<td>• The fitness value (( \text{fit}_R )) associated with the previous filtering threshold is calculated.</td>
</tr>
<tr>
<td></td>
<td>• A chromosome is copied to the FCS if its true fitness is greater than ( Rf ).</td>
</tr>
</tbody>
</table>

**Table 2.1: Summary of explicit and implicit filtering**

The overall effect of the filtering process is the continual extraction of high performance solutions from the search process without the need for apriori knowledge relating to the design model. A further advantage is the elimination of the need to maintain stable niches within the population since all solutions are extracted dynamically during search. Furthermore the scaling of the fitness range achieved by the normalisation process (figure 2.6) promotes consistent results (allowing for inaccuracies in the fitness distribution model) across a range of design domains for a given filtering vector.

**2.2.3 The Final Clustering Set (FCS)**

The FCS may be thought of as a database that stores all high performance solutions extracted during search. The FCS therefore holds all information used by the designer to
assess and explore the design domain. Since the FCS is also used throughout this thesis to assess the performance of each presented COGA technique it is vital that its nature and growth during search is fully explored. To demonstrate this, two examples are presented, the first demonstrates the growth of the FCS during search. The second looks at the effect of $R_f$ upon the FCS.

2.2.3.1 The Modified Himmelblau Function

The modified Himmelblau function [Beasely et. al. 1993] is a two dimensional function of the form,

$$z(x, y) = 200 - (x^2 + y - 11)^2 - (x + y^2 - 7)^2$$

...2-8

The function consists of four optima of approximately equal magnitude, the optima located at $x = -4$, $y = -3.5$ is more isolated and sensitive\(^2\) than the remaining peaks and is therefore less attractive during search. Two ridges (figures 2.7 and 2.8) interconnect the three remaining peaks. Although the Himmelblau function is a simple one, it is useful in this context since it is straightforward to visualise the function and the subsequent FCS solutions superimposed upon it.

\(^2\) In this context, sensitivity refers to the change in objective function fitness for small input perturbations.
2.2.3.2 Dynamic Growth of the FCS

Each point within figures 2.9 to 2.13 represents a solution present in the FCS at the end of each of the five search stages. The fitness of each solution is denoted by the colour of the point. The following COGA parameters were used during search.

\[ g = \{5, 10, 15, 20, 25\} \]
\[ m = \{0.08, 0.06, 0.04, 0.02, 0.01\} \]
\[ R_f = \{0.0, 0.0, 0.0, 0.0, 0.0\} \]

At the end of the first search stage (figure 2.9) the FCS consists of large proportions of low performance solutions, however even at this early stage there are solutions located at all four peaks. As the next search stage commences the mutation probability drops and implicit filtering commences, this results in an increase in both FCS size and average fitness (figure 2.10). As the mutation probability drops further in the third, fourth and fifth search stages, search continues to converge upon the optimal regions. Consequently larger proportions of solutions with optimal or near optimal fitness pass into the FCS. This increase in set cover about the optimal regions at later stages of search is not evident in the most sensitive high performance region \((x = -4, y = -3.5)\).

![Figure 2.9: FCS after search stage 1](image1)

![Figure 2.10: FCS after search stage 2](image2)
This suggests that solutions located at this peak are quickly eliminated from the gene pool as solutions from the more robust regions dominate the evolutionary process. Unlike MMO, this inability to maintain stable niches is of lesser importance with COGA since solutions located about this niche have been extracted at earlier stages of search and are therefore present in the FCS.

Figure 2.11: FCS after search stage 3

Figure 2.12: FCS after search stage 4

Figure 2.13: FCS after search stage 5
   (Colour key is common to all five plots)

It can be seen that the growth of the FCS is a highly dynamic process. During the early stages of search global information of relatively low fitness enters the FCS. As search converges solutions with higher fitness, located at or about optimal regions generally enter the FCS. It has been shown [Parmee 1996a] that set cover at later stage of search may be
increased further by reducing the corresponding filtering thresholds. However this practice should be exercised with caution since it may introduce low performance solutions into the FCS.

2.2.3.3 The Filtering Vector

The effects Rf has upon the nature of the FCS is investigated by executing four COGA runs (using the same initial population) using increasingly discriminatory filtering. The two dimensional test function consists of seven optimal regions (figure 2.14 & 2.15), and is of the form,

\[ z(x, y) = \sqrt{a + b} + \frac{\sin(2x)\cos(2y)}{20} \]

where,

\[ a = \frac{1}{1 + e^{-\frac{xy}{2}}} \]
\[ b = \frac{(\cos(x) + 1)(\cos(y) + 1)}{2} \]

Figure 2.14: Surface plot of the seven peak function
Figure 2.15: Contour plot of the seven peak function
(All contours below 0.5 omitted)

If relatively low filtering is used (figure 2.16), FCS size is maximised since more solutions pass filtering, however there is also an increase in the number of low performance
solutions in the FCS. As $R_f$ is increased (figure 2.17) the size of the FCS decreases as the low performance solutions fail to pass the filtering process, the average fitness of the FCS consequently increases. If $R_f$ is increased further (figure 2.18), the solutions located at the peripheries of the optima are eliminated, as the standard deviation of the FCS diminishes. In the final plot (figure 2.19) the large majority of solutions populating the high performance peaks are eliminated, with and only a handful of optimal solutions remaining.

**Figure 2.16:** $R_f = \{0.0, 0.0, 0.0, 0.0, 0.0\}$  
**Figure 2.17:** $R_f = \{0.5, 0.5, 0.5, 0.5, 0.5\}$

**Figure 2.18:** $R_f = \{1.0, 1.0, 1.0, 1.0, 1.0\}$  
**Figure 2.19:** $R_f = \{1.5, 1.5, 1.5, 1.5, 1.5\}$  
(Colour key is common to all four plots)

It follows that the filtering threshold controls the number and average fitness of solutions copied to the FCS. Higher filtering thresholds are more discriminatory allowing only the
fittest solutions to pass to the clustering set. Conversely, low filtering thresholds promote maximum set cover but reduce the average fitness of FCS. An interesting point to note is that at the largest values of $R_f$, only optimal solutions remain in the FCS and in this special case COGA acts in a similar manner to the MMO algorithms presented in section 2.1.1.
CHAPTER 3

COGA and the Support of Conceptual Engineering Design

3.1 Introduction

This chapter investigates the role COGA plays in supporting the conceptual design phase by demonstrating its application to four commonly encountered design scenarios described by real world models. The examples are,

- Single level model – single objective decomposition.
- Mixed parameter model – single objective decomposition.
- Single level model – multi objective decomposition.
- Single level model – infeasible region decomposition.

The vmCOGA algorithm of section 2.2 is used throughout the following study. The generic appeal of COGA is further evident when it is considered that the only difference between the COGA algorithm used here and that used in the 2d FCS studies of sections 2.2.3.2 and 2.2.3.3 is an increase in the generation vector, \( m = \{50, 100, 150, 200, 250\} \). This reflects the higher dimensionality of the design models and the subsequent increase in design options. Two models are investigated, miniCAPS and TBCOM.

3.1.1 MiniCAPS

MiniCAPS is a simplified version of the British Aerospace CAPS model (Computer Aided Project Studies) which is used to support conceptual airframe design [Webb 1997]. MiniCAPS consists of eight continuous input variables (table 3.1) and twelve continuous output variables (table 3.2). The model contains a number of subroutines, each of which calculates properties relating to criteria such as performance, wing geometry, propulsion,
fuel capacity, structural integrity and so on. During this thesis miniCAPS is used for both single and multi-objective decomposition.

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit of Measure</th>
<th>Minimum Value</th>
<th>Maximum Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Climb Mach Number</td>
<td>-</td>
<td>0.4</td>
<td>0.9</td>
</tr>
<tr>
<td>Cruise Height</td>
<td>m</td>
<td>5000.0</td>
<td>12000.0</td>
</tr>
<tr>
<td>Cruise Mach number</td>
<td>-</td>
<td>0.6</td>
<td>0.9</td>
</tr>
<tr>
<td>Gross Wing Plan Area</td>
<td>m²</td>
<td>20.0</td>
<td>80.0</td>
</tr>
<tr>
<td>Wing Aspect Ratio</td>
<td>-</td>
<td>1.5</td>
<td>6.0</td>
</tr>
<tr>
<td>Wing Taper Ratio</td>
<td>-</td>
<td>0.1</td>
<td>0.6</td>
</tr>
<tr>
<td>Wing Leading Edge Sweep</td>
<td>°</td>
<td>25.0</td>
<td>60.0</td>
</tr>
<tr>
<td>Wing Thickness / Chord Ratio</td>
<td>-</td>
<td>0.03</td>
<td>0.12</td>
</tr>
</tbody>
</table>

Table 3.1: MiniCAPS Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Abbreviation</th>
<th>Units of Measure</th>
<th>Optimisation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Take Off Distance</td>
<td>-</td>
<td>m</td>
<td>Minimise</td>
</tr>
<tr>
<td>Landing Speed</td>
<td>-</td>
<td>m/s²</td>
<td>Minimise</td>
</tr>
<tr>
<td>Subsonic Specific Excess Power</td>
<td>SEP 1</td>
<td>N</td>
<td>Maximise</td>
</tr>
<tr>
<td>Supersonic Specific Excess Power</td>
<td>SEP 2</td>
<td>N</td>
<td>Maximise</td>
</tr>
<tr>
<td>Subsonic Attained Turn Rate</td>
<td>ATR 1</td>
<td>°s⁻¹</td>
<td>Maximise</td>
</tr>
<tr>
<td>Supersonic Attained Turn Rate</td>
<td>ATR 2</td>
<td>°s⁻¹</td>
<td>Maximise</td>
</tr>
<tr>
<td>Subsonic Sustained Turn Rate</td>
<td>STR 1</td>
<td>°s⁻¹</td>
<td>Maximise</td>
</tr>
<tr>
<td>Supersonic Sustained Turn Rate</td>
<td>STR 2</td>
<td>°s⁻¹</td>
<td>Maximise</td>
</tr>
<tr>
<td>Ferry Range</td>
<td>-</td>
<td>km</td>
<td>Maximise</td>
</tr>
<tr>
<td>Take-Off Mass</td>
<td>-</td>
<td>kg</td>
<td>Minimise</td>
</tr>
<tr>
<td>Wing Span</td>
<td>-</td>
<td>m</td>
<td>-</td>
</tr>
<tr>
<td>Wing Chord / Fuselage Length</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 3.2: MiniCAPS Outputs

3.1.2 TBCOM

The performance of a gas turbine increases as its operating temperature increases [Cohen 1987]. In order to maintain engine integrity, the temperature of the turbine blades must be reduced during operation. This is achieved by passing a quantity of cooler air over the surface of both the turbine blade and blade root. This cooling air is “bled” from the turbine and therefore reduces the performance of the engine. Consequently, the overall objective is to minimise the amount of cooling required whilst maintaining the turbine blade temperatures within safe operating ranges. The Rolls Royce Turbine Blade Cooling Model (TBCOM) was developed by Rolls Royce and the PEDC to minimise coolant mass flow rate inside a series of cooling hole geometries within a gas-turbine blade [Roy 1996]. A
single discrete variable (geometry) relates to three internal cooling hole geometries (plane, ribbed and pedestal) and controls the operating ranges of two of the eleven continuous input variables (heat transfer coefficient factor and coefficient of discharge). The remaining eight input ranges are independent of geometry configuration (table 3.3). In this example TBCOM is used to minimise radial coolant mass flow rate whilst maintaining three other outputs within acceptable performance bounds (table 3.4).

<table>
<thead>
<tr>
<th>Name</th>
<th>Abbreviation</th>
<th>Unit of Measure</th>
<th>Minimum Value</th>
<th>Maximum Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geometry (Plane)</td>
<td>Geom1</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Geometry (Ribbed)</td>
<td>Geom2</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Geometry (Pedestal)</td>
<td>Geom3</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Radial Heat Transfer Coefficient – Geometry 1</td>
<td>Fhc1</td>
<td>-</td>
<td>1.0</td>
<td>1.6</td>
</tr>
<tr>
<td>Radial Heat Transfer Coefficient – Geometry 2</td>
<td>Fhc2</td>
<td>-</td>
<td>1.3</td>
<td>3.0</td>
</tr>
<tr>
<td>Radial Heat Transfer Coefficient – Geometry 3</td>
<td>Fhc3</td>
<td>-</td>
<td>1.8</td>
<td>3.2</td>
</tr>
<tr>
<td>Radial Coefficient of Discharge – Geometry 1</td>
<td>Cdr1</td>
<td>-</td>
<td>0.6</td>
<td>0.75</td>
</tr>
<tr>
<td>Radial Coefficient of Discharge – Geometry 2</td>
<td>Cdr2</td>
<td>-</td>
<td>0.4</td>
<td>0.6</td>
</tr>
<tr>
<td>Radial Coefficient of Discharge – Geometry 3</td>
<td>Cdr3</td>
<td>-</td>
<td>0.2</td>
<td>0.4</td>
</tr>
<tr>
<td>Inlet Temperature</td>
<td>Tcl</td>
<td>K</td>
<td>700.0</td>
<td>800.0</td>
</tr>
<tr>
<td>Wall Thickness</td>
<td>dh</td>
<td>m</td>
<td>0.00075</td>
<td>0.0025</td>
</tr>
<tr>
<td>Blade Thermal Conductivity</td>
<td>kw</td>
<td>W Km&lt;sup&gt;-1&lt;/sup&gt;</td>
<td>18.0</td>
<td>33.0</td>
</tr>
<tr>
<td>Inlet / Outlet Pressure Ratio</td>
<td>Rp</td>
<td>-</td>
<td>1.05</td>
<td>1.6</td>
</tr>
<tr>
<td>Radial Passage Perimeter Ratio</td>
<td>Rs</td>
<td>-</td>
<td>0.5</td>
<td>1.5</td>
</tr>
<tr>
<td>Cooling Hole Diameter</td>
<td>df</td>
<td>m</td>
<td>0.00001</td>
<td>0.0004</td>
</tr>
<tr>
<td>Cooling Hole Coefficient of Discharge</td>
<td>Cdf</td>
<td>-</td>
<td>0.6</td>
<td>0.75</td>
</tr>
<tr>
<td>Cooling Heat Transfer Coefficient Factor</td>
<td>Ff</td>
<td>-</td>
<td>1.0</td>
<td>1.6</td>
</tr>
<tr>
<td>Cooling Hole Pressure Ratio</td>
<td>Rpf</td>
<td>-</td>
<td>0.2</td>
<td>0.4</td>
</tr>
</tbody>
</table>

Table 3.3: TBCOM Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Abbreviation</th>
<th>Units of Measure</th>
<th>Optimisation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Radial Coolant Mass Flow Rate</td>
<td>Wcr</td>
<td>kgs&lt;sup&gt;-1&lt;/sup&gt;</td>
<td>Minimise</td>
</tr>
<tr>
<td>Film Hole Coolant Mass Flow Rate</td>
<td>Wcf</td>
<td>kgs&lt;sup&gt;-1&lt;/sup&gt;</td>
<td>Constrained</td>
</tr>
<tr>
<td>Gas Side Wall Temperature</td>
<td>Twg</td>
<td>K</td>
<td>Constrained</td>
</tr>
<tr>
<td>Film Side Wall Temperature</td>
<td>Twf</td>
<td>K</td>
<td>Constrained</td>
</tr>
</tbody>
</table>

Table 3.4: TBCOM Outputs

### 3.2 Single Objective Decomposition: MiniCAPS Model

The single objective decomposition investigation presented here follows on from the Rf investigation of section 2.2.3.3 by showing how COGA may be used within a real-world design domain to investigate different regions of the search space. Four independent COGA runs are executed upon miniCAPS using increasingly discriminatory filtering.

48
thresholds. In each instance all eight input parameters are optimised, however to simplify visualisation hyperplanes representing two of the eight variables are presented to the designer. In this example Landing Speed is minimised, colour is used to depict the fitness of each solution.

Figure 3.1: Minimising Landing Speed  
\[ R_f = \{1.0, 1.0, 1.0, 1.0, 1.0\} \]

Figure 3.2: Minimising Landing Speed  
\[ R_f = \{1.5, 1.5, 1.5, 1.5, 1.5\} \]

Figure 3.3: Minimising Landing Speed  
\[ R_f = \{2.0, 2.0, 2.0, 2.0, 2.0\} \]

Figure 3.4: Minimising Landing Speed  
\[ R_f = \{2.5, 2.5, 2.5, 2.5, 2.5\} \]  
(Colour key is common to all four plots)

The statistical composition of the FCS for each of the four runs is shown in table 2.5. Column two gives the size of the FCS. Columns three and four give the average and standard deviation of the FCS fitness. The fitness of the “best” solution found during search is shown in column five.
Figures 3.1 – 3.4 present wing thickness-to-chord ratio against wing leading edge sweep for each of the COGA runs. Results given by the runs using relatively low filtering vectors (figures 3.1 & 3.2) indicate the presence of a curved ridge of solutions arcing from the top left to the lower right portions of the hyperplane. As filtering becomes increasingly discriminatory, this ridge further decomposes (figure 3.3) into two regions of near optimal performance located in the upper left and lower right sections of figure 3.4.

The power of the COGA approach is apparent. During preliminary runs where domain knowledge is limited low $R_f$ values give maximum information relating to the overall nature of the search space. As domain knowledge and confidence increase, more discriminatory filtering presents succinct regions to the designer. Further information such as cost of material, unit cost and designer expertise may then be used to assess the merits of each region, indicating the viability of future design directions. Once this has been achieved, subsequent search may be constrained within the identified regions, eliminating the unnecessary investigation of the low performance regions of the design space. This effectively increases the resolution of the design variable (through a reduction in its range) and may identify further areas of interest within the decomposed regions.

### Table 3.5: Final clustering set composition

<table>
<thead>
<tr>
<th>Figure</th>
<th>Number</th>
<th>ave (ms$^{-1}$)</th>
<th>sd (ms$^{-1}$)</th>
<th>best (ms$^{-1}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>4006</td>
<td>66.67</td>
<td>1.87</td>
<td>59.31</td>
</tr>
<tr>
<td>3.2</td>
<td>1317</td>
<td>62.50</td>
<td>1.21</td>
<td>57.37</td>
</tr>
<tr>
<td>3.3</td>
<td>156</td>
<td>60.80</td>
<td>1.11</td>
<td>57.50</td>
</tr>
<tr>
<td>3.4</td>
<td>14</td>
<td>59.56</td>
<td>0.59</td>
<td>58.50</td>
</tr>
</tbody>
</table>

### 3.3 Mixed Parameter Single Objective Decomposition: TBCOM Model

In practice, design domains are often described by mixed parameter design models where discrete parameters representing individual design configurations relate to independent regions of the search space and may define individual fitness landscapes. When
investigating mixed parameter models, it is important to identify representative results for each discrete parameter since a configuration deemed "fit" in the optimised parameter, may be less fit in other performance criteria.

**Figure 3.5:** FCS taken from a COGA run upon the TBCOM model

**Figure 3.6:** Idealised high performance boundaries for the results shown in figure 3.5

The mixture of discrete and continuous variables is encoded using a structured approach (section 1.5.1.1). Fitness scaling is used to encourage the generation and extraction of equal amounts of geometry one, two and three solutions in the FCS. The generation and mutation vectors are the same as those used in the miniCAPS study. To simplify graphical representation, the size of the clustering set is reduced by applying a more discriminatory filtering vector, $R_f = \{1.5, 1.5, 1.5, 1.5, 1.5\}$. The two dimensional slice of radial passage perimeter ratio against inlet / outlet pressure ratio in figure 3.5 clearly shows three distinct regions relating to each geometry configuration. If the boundary of each high performance region is idealised by connecting the outermost points (figure 3.6), the regions become more visible (in practice lower filtering thresholds would increase the size of the clustering set and reduce possible misinterpretation when idealising the regional boundaries). Arguably the most important feature of figure 3.6 is the emergence of common regions where high performance is achieved irrespective of internal geometry. It is unlikely that such a common region would be a forthcoming if a more traditional optimisation algorithm
were used. As with the single objective example, the designer may use the information inherent within the solutions of these regions and further in-house and personal knowledge, to assess the design potential of each design configuration.

3.4 Multi-Objective Decomposition: MiniCAPS Model

More usually, the fitness of a solution is assessed in terms of several, often-conflicting criteria. To determine the multi-objective solution, “trade-offs” or compromises must be made between objectives. When using techniques such as weighted sum [Ben-Tal 1979] or Lexicographic [Hwang et al. 1979] the relative importance of each objective must be known on an apriori basis. However, during conceptual design, there may be insufficient knowledge to accurately predict such weightings, which may also change during the design process. COGA aids the multi-objective decomposition process by operating in a sequential manner where \( n \) independent runs are executed and in each case a different objective is optimised. The contents of each FCS may then be superimposed within a hyperplane, and by relaxing the filtering vector, common or mutually inclusive regions may be interpreted as areas of multi-objective compromise. Figures 3.7 – 3.9 show the results of three independent \( \text{vmCOGA} \) runs upon the miniCAPS model. In each instance a difference objective has been optimised (Ferry Range, SEPI and ATR1

\[
R_{\text{Range}} = \{1.0, 1.0, 1.0, 1.0, 1.0\}
\]

\[
R_{\text{ATRI}} = \{1.0, 1.0, 1.0, 1.0, 1.0\}
\]
The objectives are highly conflicting since each high fitness region relates to differing areas within the wing aspect ratio, gross wing plan area hyperplane. When Ferry Range is maximised, large wing areas and high aspect ratios (indicative of glider type aircraft) promote maximum lift. The antithesis is noted when SEP is maximised, here small wing areas and low aspect ratios minimise aircraft weight. To determine a solution which best satisfies all three objectives, a compromise region must be sought.

Figure 3.10 shows the idealised high performance boundary for each objective superimposed onto a single hyperplane. In the case of Range and ATR there is a mutually inclusive region, which may be interpreted as an area where a solution would satisfy both range and ATR objectives, however this common region does not include solutions relating to the SEP objective.

If $Rf_{SEP}$ is isolated and relaxed and a further COGA run executed it can be seen that the high performance boundary relating to SEP moves towards the compromise region for range and ATR (figure 3.11).
The further reduction of $Rf_{SEP}$ results in a compromise region for all three objective combinations (figure 3.12 – 3.14). In a practical sense the relaxing of $Rf_{SEP}$ is analogous to reducing the importance of the SEP objective since weaker solutions are allowed to pass filtering and enter the FCS. Using this technique, the designer may explore the relationship and interactions between objectives in the variable space and discover areas
where more directed multi-objective optimisation techniques may be constrained to search for the optimal compromise solution.

![Diagram](image)

**Figure 3.14:** Compromise region \( SEP \cap ATR \cap Range \)

\[
R_{\text{Range}} = \{1.0, 1.0, 1.0, 1.0, 1.0\}
\]

\[
R_{\text{ATR}} = \{1.0, 1.0, 1.0, 1.0, 1.0\}
\]

\[
R_{\text{SEP}} = \{0.0, 0.0, 0.0, 0.0, 0.0\}
\]

### 3.5 Infeasible Region Decomposition: TBCOM Model

The COGA applications presented in the preceding sections have all dealt with the optimisation of one or more of the design model objectives. Once completed, the designer uses the information within the FCS to constrain search within one or more of the decomposed high performance regions. However, it may be argued that it is equally important to determine where not to search, in other words, to decompose the design space into regions of infeasibility defined by the constraint models. One may wrongly assume that such regions are represented by the "empty space" in the decomposed hyperplanes (figure 3.15 presents two such regions in the top left and bottom right of the \( Rs, Rp \) hyperplane), however this is not the case. Consider a solution that fails to enter the FCS, such a region although of poor objective function fitness may be fit in other criteria and therefore offers considerable design potential. It is therefore false to assume that such solutions populate infeasible regions.
There are two approaches that are commonly used to find optimal solutions within constrained search spaces, modification of the genetic operators and penalising chromosomes which fail to meet the constraints [Richardson et. al. 1989]. As is the case with the majority of optimisation techniques, both of these approaches give very little information relating to the nature of the constrained regions. If COGA is used to maximise constraint violation, the subsequent decomposed high performance regions illustrate areas within the search space where the penalty functions are most active, and subsequently define infeasible regions of the search space.

TBCOM uses the second constraint handling technique (the penalisation of chromosomes that violate the constraints) to maintain the operating temperature of the turbine blades within acceptable operating bounds. The three TBCOM constraints are shown in table 3.6. The amount, if any, that each constrained variable violates the constraint boundaries are denoted by \( v_1 \), \( v_2 \) and \( v_3 \) respectively. If any one of the violation factors \( p_1 \), \( p_2 \) and \( p_3 \) are less than 1, the fitness of a solution is reduced according to,

\[
\text{fitness} = \frac{1}{10Wcr} (p_1 \times p_2 \times p_3) \quad \ldots 3-1
\]

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Operating Bound</th>
<th>Constraint Violation</th>
<th>Penalty Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( 1200.0K \leq Twg \leq 1300.0K )</td>
<td>( \begin{align*} v_1 \left{ Twg - 1300.0 &amp; \quad Twg &gt; 1300.0K \ 1200.0 - Twg &amp; \quad Twg &lt; 1200.0K \ 0 &amp; \quad \text{otherwise} \end{align*} )</td>
<td>( p_1 \left{ \begin{align*} 0 &amp; \quad v_1 &gt; 150.0K \ 1 - \frac{v_1}{150} &amp; \quad \text{otherwise} \end{align*} )</td>
</tr>
<tr>
<td>2</td>
<td>( Twf \leq 1300.0K )</td>
<td>( \begin{align*} v_2 \left{ Twf - 300.0 &amp; \quad Twf &gt; 1300.0K \ 0 &amp; \quad \text{otherwise} \end{align*} )</td>
<td>( p_2 \left{ \begin{align*} 0 &amp; \quad v_2 &gt; 150.0K \ 1 - \frac{v_2}{150} &amp; \quad \text{otherwise} \end{align*} )</td>
</tr>
<tr>
<td>3</td>
<td>( \frac{Wcr}{Wcf} \geq 0.8 )</td>
<td>( \begin{align*} v_3 \left{ 0.8 - \frac{Wcr}{Wcf} &amp; \quad Wcr &lt; 0.8 \ \frac{Wcr}{Wcf} &amp; \quad \text{otherwise} \end{align*} )</td>
<td>( p_3 \left{ \begin{align*} 0 &amp; \quad v_3 &gt; 0.4 \ 1 - \frac{v_3}{0.4} &amp; \quad \text{otherwise} \end{align*} )</td>
</tr>
</tbody>
</table>

Table 3.6: TBCOM Constraint Model
If the results of a COGA run (minimising Wcr and subject to the constraint model) are presented a distinct high performance region is evident (figure 3.15). This suggests that optimal flow rates are achieved with low values for both Rs and Rp. The under populated upper left and lower right regions of the hyperplane may be indicative of either regional low performance or may be due to action of the constraint equations.

![Figure 3.15: Minimise Wcr (Subject to the constraint equations)
Rf = {0.0, 0.0, 0.0, 0.0, 0.0}
](image)

To determine the infeasible regions, further COGA runs are executed in each instance one of the three possible constraint violations v1, v2 or v3 is optimised. The decomposed regions relating to v1 and v3 are shown in figures 3.16 and 3.17 respectively. The second constraint is redundant since no examples are found that violate the constraint boundary, suggesting that the constraint should either be redefined or removed from the design model. The region relating to v1 is located at the lower right of the Rp / Rs hyperplane, whilst the constrained region for v3 is located at larger values of Rs. It is clear from the plots that both regions concur with one of the underpopulated regions of figure 3.15. However, the region located at the top left of the Rs / Rp hyperplane was not identified in either v1, v2 or v3 optimisation, suggesting that the under population of the region noted in figure 3.15 is due to general low performance and not the action of any constraint equation.
Within real world design environments, COGAs are used in an iterative manner where rapid sequential runs allow the designer to “walk through” the design space gaining global knowledge relating to the overall nature of the design domain and more specific local knowledge relating to individual high performance regions. However the ability of COGA to efficiently support such a design environment, is heavily dependent upon the link or interface between the designer and the COGA technique. Such a link should allow the designer to rapidly identify multiple high performance regions and to extract relevant design information relating to each region. Extracted information should be presented in a manner so as to provide sufficient design information to allow the designer to assess possible design directions whilst avoiding information overload. Supported by extracted information, the designer may execute further runs where parameter bounds, objectives and filtering thresholds are modified to further extend design domain knowledge and support the identification of novel design areas.

Collaborative work with Rolls Royce and British Aerospace Military Aircraft has addressed this by developing a Graphical User Interface (GUI) which provides an interface between the designer, COGA, miniCAPS and TBCOM. The GUI, developed by Parmee,
Bonham and Richter is both UNIX and Windows 95 based, and is written in C++ using the Borland development environment Borland C++ Builder v3.0.

The GUI uses a pull-down menu environment where the user may directly set, monitor and control all COGA search parameters in addition to objectives and variable bounds relating to the problem domain (figures 3.18 and 3.19).

<table>
<thead>
<tr>
<th>Variable Ranges</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Climb Mach number</td>
</tr>
<tr>
<td>Cruise height</td>
</tr>
<tr>
<td>Cruise Mach number</td>
</tr>
<tr>
<td>Wing area</td>
</tr>
<tr>
<td>Wing aspect ratio</td>
</tr>
<tr>
<td>Wing taper ratio</td>
</tr>
<tr>
<td>Wing sweep</td>
</tr>
<tr>
<td>Wing t/c ratio</td>
</tr>
<tr>
<td>Bypass ratio</td>
</tr>
</tbody>
</table>

Objective to optimise: Ferry range

Figure 3.18: Defining the search space and optimised objective

<table>
<thead>
<tr>
<th>Edit Stages</th>
</tr>
</thead>
<tbody>
<tr>
<td>p(mutation)</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
</tbody>
</table>

Figure 3.19: Defining the COGA parameters $g$, $m$ and $R_f$
Once search is completed, the size of the FCS along with the average, minimum and maximum fitness of the solutions contained within it are presented (figure 3.20). If during the first filtering stage, there are no solutions that pass filtering a warning and a recommendation to relax $R_f$ is displayed (figure 3.21).

![Debug Info](image1)

**Figure 3.20:** FCS analysis presented at the end of search

![Error](image2)

**Figure 3.21:** Warning displayed when no solutions pass the filtering process

![2D Select](image3)

**Figure 3.22:** Hyperplane selection
If there are sufficient solutions within the FCS, the GUI prompts the user to specify which hyperplanes to plot (figure 3.22), the corresponding results are plotted using colour to represent the fitness of each solution (figure 3.23).

Figure 3.23: 2d scatter plot showing Wing Aspect Ratio and Gross Wing Plan Area

<table>
<thead>
<tr>
<th>Solution Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ferry range</td>
</tr>
<tr>
<td>Climb Mach number</td>
</tr>
<tr>
<td>Cruise height</td>
</tr>
<tr>
<td>Cruise Mach number</td>
</tr>
<tr>
<td>Wing area</td>
</tr>
<tr>
<td>Wing aspect ratio</td>
</tr>
<tr>
<td>Wing taper ratio</td>
</tr>
<tr>
<td>Wing sweep</td>
</tr>
<tr>
<td>Wing l/c ratio</td>
</tr>
<tr>
<td>Bypass ratio</td>
</tr>
</tbody>
</table>

Figure 3.24: Examining individual design solutions
The fitness and remaining design parameters relating to any solution can be obtained by selecting the desired solution with the mouse (figure 3.24).

Even from this limited overview of the operation of the GUI, the power of such a tool is evident. The GUI allows for the rapid and efficient realisation of all of the possible COGA applications, this is achieved within a user-friendly framework and allows for the online exploration and subsequent decomposition of the design domain. The facility to save and load results and COGA configurations allows the designer to investigate many different design directions and to reinvestigate them as and when required.

3.7 The Interactive Evolutionary Design Station

The long term goal of COGA is its integration into the Interactive Evolutionary Design Station (IEDS) [Parmee et. al. 1998, 1999a & Parmee 1999c]. The IEDS is an online conceptual design support tool that contains a number of interrelated elements (figure 3.25). The core of the IEDS is a group of co-evolutionary components that simultaneously investigate differing aspects of the design problem [Parmee et. al. 1999b]. The use of parallel evolution further increases the speed of search. Intelligent agents and rule-based preferences provide the communication medium between processes in terms such as state recognition, degree of convergence, constraint satisfaction etc. [Cvetkovic et. al. 1999]. Interaction between the co-evolutionary processes may be achieved by either automatic modification of the rule-based preferences or controlled directly by the designer. Throughout search, information is continuously extracted by COGA and presented to the design team, thereby forming the information gathering part of the IEDS. This information may be used by the designer to support the focussing of the coevolutionary search components upon specific regions of the design domain.
The overall goal of the IEDS is to provide a dynamic exploitative medium that supports design team knowledge and stimulates continuous interaction with the design model. Throughout this process information is extracted and presented to the design team, this increases general design domain knowledge and reduces uncertainty when changes are made to the overall design concept.
CHAPTER 4

Improving the Performance of the Diverse Search Engine

4.1 Introduction

The variable mutation strategy detailed in section 2.2.1 provided a preliminary strategy for promoting diverse search within COGA. However it is questionable whether high mutation rates are the most efficient means for achieving this [Goldberg 1987, Louis et. al. 1993], since there is a significant possibility that schemata may be destroyed during search thereby reducing heritability (the passing of desirable performance characteristics from parent to child) [DeJong 1999]. Research has also shown that the setting of the mutation probability during search is by no means arbitrary [Davis 1989, Bäck 1993]. It is subsequently suggested that the value of the mutation vector \( m \), may also have a tangible effect upon performance since it can not be assumed that the universal vector \{0.08, 0.06, 0.04, 0.02, 0.01\} will promote generic diversity across any design domain. Finally, without the presence of further diversity operators (such as Sharing or Crowding), mutation alone may be unable to maintain isolated or sensitive peaks within the population. As the mutation probability drops at later stages of search, the possibility of one peak dominating the population is high. This eventuality has been demonstrated empirically in the FCS investigation of section 2.2.3.2.

The following sections explore the possibility of replacing variable mutation with a series of existing search algorithms that use more specialised diversity promoting mechanisms.
4.2 The Test Algorithms

Maintaining the correct degree of diversity within a population is vital in order to achieve the optimum balance between exploration and exploitation (section 2.2.1). If the degree of diversity is too high search may degenerate into a random walk where further local investigation of identified high performance regions will not occur. If the degree of diversity is insufficient, sampling of the design space reduces and search rapidly converges to a single, often sub optimal peak.

Historically, the problem of maintaining diversity during search may be tackled in a number of ways. MMO mechanisms such as Crowding and Sharing have been discussed in section 2.1.1, further techniques include the modification of either the crossover operator [Brooker 1987, Syswerda 1989, Eshelman 1991] or the selection scheme [Baker 1987, Goldberg 1989]. However maintaining diversity is only one of the required criteria for any prospective COGA search engine (section 2.1), the algorithm must also be insensitive to the effects of genetic drift (small changes in GA convergence in successive runs caused by the stochastic nature of search). Furthermore the degree of apriori knowledge required to tune the algorithm should be minimised since the generic application of such algorithms should be anticipated. Techniques such as Fitness Sharing and DeJong Crowding are therefore inappropriate since parameters such as $\alpha$, $\sigma_{\text{share}}$ and $CF$ are dependent upon peak shape and location and therefore vary for any given search space.

Three algorithms are investigated and compared with the variable mutation strategy of section 2.2.1, Deterministic Crowding (DC), the ECOGA and Modified Uniform Crossover coupled with Incest Prevention (HUX / IP). It should be noted that the selection of these algorithms is by no means arbitrary. Research suggests that all three approaches promote increased diversity within the population [Mahfoud 1992, Davidor 1991, Davidor 65].
et. al. 1993 & Eshelman et. al. 1991]. Furthermore the need to tune algorithm parameters before search is not required in both the DC and ECOGA algorithms. HUX / IP does require some apriori tuning in order to set the incest thresholds during search, to maintain consistency such values are generic in all test cases (section 4.2.2). The following sections introduce the test algorithms. Deterministic Crowding, the ECOGA and vmCOGA have been discussed in sections 2.1.1.1, 2.1.1.5 and 2.2 respectively and only a cursory review is given here.

### 4.2.1 vmCOGA

The vmCOGA algorithm of section 2.2 forms the benchmark or control algorithm used throughout the thesis. To maintain consistency with previous work [Parmee 1996b], the generation, mutation and filtering vectors used during search are,

\[ g = \{5, 10, 15, 20, 25\} \]

\[ m = \{0.08, 0.06, 0.04, 0.02, 0.01\} \]

\[ Rf = \{0.75, 0.75, 0.5, 0.25, 0.00\} \]

### 4.2.2 Modified Uniform Crossover and Incest Prevention (HUX / IP)

Eshelman uses incest prevention (IP) and a variant of uniform crossover (HUX) within the CHC algorithm to promote high levels of diversity during search [Eshelman 1991]. Incest prevention allows parents to reproduce only if a genotypic distance metric between them is greater than a pre-defined incest threshold. This reduces the probability of crossover between siblings and their subsequent premature propagation throughout the population. The incest threshold is reduced during search in proportion to the ease of children surviving into the next generation. HUX is a variant of uniform crossover [Syswerda 1989] where exactly half of the bits are randomly swapped. Eshelman argues that HUX / IP alone can not guarantee the elimination of premature convergence and some mutation mechanism should be used. This is achieved by restarts; upon stagnation the fittest
chromosome is used as a template and the population re-initialising by randomly flipping 35% of the template bits. However the small runtimes associated with COGA clearly make this impractical, hence, mutation (set to 0.01) is used. The incest threshold is initially set to half the bit length of the chromosome (this is twice the value suggested by Eshelman and further promotes diversity) and is subsequently halved at each filtering stage. The number of crossover operations per generation matches that during a COGA run. If a selected pair of parents do not meet the incest threshold further parents are chosen until the required number of crossover operations per generation is met.

4.2.3 Deterministic Crowding

The primary diversity mechanism within DC is the replacement of the most similar. The selection scheme serves only to randomly pair the population. It follows that during each generation of a DC algorithm the entire population takes part in the reproduction stage. It is acknowledged that this increase in reproductive activity when compared with the remaining test algorithms, where only 60% of the population is reproduced, should be taken into account when DC results are assessed. Throughout search, 1pt crossover and a mutation rate of 0.01 are used.

4.2.4 ECOGA

Section 2.1.1.5 indicated that the selection of the ECOGA grid size is by no means arbitrary. Furthermore, the lack of published guidelines relating to the selection of such criteria makes any grid selection questionable. The smallest grid size investigated (10 x 10) is comparable with the population sizes of the remaining test algorithms. This initial grid is then doubled and trebled to produce the 20 x 20 and 30 x 30 ECOGA grids, these relate to population sizes of 400 and 900 respectively. Throughout search uniform crossover and a mutation rate of 0.025 helps to reduce the onset of grid domination by a single high performance niche [Spiessens 1991]. Since the ECOGA is a steady state GA,
each explicit filtering stage occurs after a "vmCOGA equivalent" number of crossover operations has taken place.

It is interesting to note the contrasting approaches used by the HUX / IP and DC algorithms to maintain diversity. Deterministic Crowding maintains diversity by promoting reproduction amongst similar individuals whereas HUX / IP forces diversity by only permitting reproduction between dissimilar individuals.

4.3 Performance Metrics

Assessing the performance of single optima seeking GAs is a relatively straightforward process. The most widely used performance measures relate to the magnitude of the optimal fitness found during search. More usually such results are averaged over a number of independent trials to increase accuracy. A more sophisticated approach was proposed by DeJong, who made a distinction between on and off line performance [DeJong 1975]. Offline performance relates to the ability of a test algorithm to determine the optimal solution irrespective of overall search time. Online performance measures the ability of an algorithm to reach an acceptable level of performance within the quickest period of time. Recent research relating to MMO algorithms has generated a new set of performance criteria, these include: a measure of the ability to maintain niches during search [Goldberg 1987]; the number of function calls to locate all optima [Mahfoud 1992]; the number of independent runs required to locate all optima [Beasely et. al. 1993] and a comparison with the expected population proportion of each peak [Parmee et. al. 1997a]. Unfortunately many of the discussed techniques offer little potential when assessing the performance of algorithms designed to locate regions of high performance within a specified period of search time. Hence seven novel performance measures are proposed and outlined in table 4.1.
Table 4.1: Performance metrics (HP – High performance)

<table>
<thead>
<tr>
<th>Performance metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Set Cover</td>
<td>Total number of solutions in the FCS</td>
</tr>
<tr>
<td>2 Global Decomposition Efficiency (GDE)</td>
<td>Percentage of the FCS that lie within any high performance region</td>
</tr>
<tr>
<td>3 Regional Decomposition (RD)</td>
<td>Number of solutions in a HP region (expressed as a percentage of vmCOGA)</td>
</tr>
<tr>
<td>4 Set cover sensitivity</td>
<td>Standard deviation of set cover (taken over 100 trials)</td>
</tr>
<tr>
<td>5 GDE sensitivity</td>
<td>Standard deviation of GDE (taken over 100 trials)</td>
</tr>
<tr>
<td>6 FCS_{ave}</td>
<td>Average fitness of solutions in FCS</td>
</tr>
<tr>
<td>7 FCS_{best}</td>
<td>Optimal fitness within FCS</td>
</tr>
</tbody>
</table>

The first two measures, Set cover and Global Decomposition Efficiency (GDE) respectively measure the number of solutions in the FCS and the percentage of solutions within the FCS that lie within any of the high performance regions. These metrics assesses the ability of each test algorithm to maximise the number of solutions populating high performance regions whilst still maintaining a crisp definition of the decomposed area. It follows that both set cover and GDE should be as high as possible. Smaller GDEs are caused by the presence of large proportions of low performance solutions that do not lie within any of the high performance regions thereby reducing the crispness of the decomposed regions.

However there are instances where high GDE values may not be indicative of high algorithm performance, an example being search that has prematurely converged upon a limited number of optimal peaks. In this case GDE approaches 100%, furthermore GDE alone does not indicate in which regions search has prematurely converged upon, or more usefully, how many solutions lie within each high performance region. The third measure, Regional Decomposition (RD) overcomes this by measuring the number of solutions within the FCS that lie within each high performance region. RD is expressed as a percentage of the corresponding value given by the vmCOGA (control) run,
RD = \left( \frac{RH - RH_{vmCOGA}}{RH_{vmCOGA}} \right) \times 100

Where,

\( RH \) = The number of solutions within the FCS that lie inside a given HP region

\( RH_{vmCOGA} \) = The number of solutions within the FCS that lie inside a given HP region (taken during a vmCOGA run)

It follows that RD of 100% indicates twice the number of solutions in a given region when compared with vmCOGA. Similarly a value of -50% indicates half the number of solutions given by vmCOGA. It is apparent that RD is dependent upon both set cover and GDE. If GDE is low and set cover sufficiently high, RD results may indicate high regional decomposition efficiency, however the large proportion of the FCS lying outside any of the high performance regions should also be taken into account.

As their names suggest, measures four (Set cover sensitivity) and five (GDE sensitivity) determine the sensitivity of each algorithm to the effects of genetic drift and noise. These are attained by calculating the standard deviation of both GDE and set cover measured over 100 trials. High standard deviations are caused by large changes in the measured values and indicate highly sensitive algorithms.

The degree of convergence within the FCS may be further assessed by calculating the average fitness of the solutions contained within it (FCS\text{ave}). Higher FCS\text{ave} values indicate higher levels of convergence. FCS\text{ave} may also support GDE and RD to predict the presence of premature convergence. The seventh and final performance metric is the more common measure of the optimal fitness within the FCS (FCS\text{best}). Although the measure is
of little use when assessing decomposition performance, it may be used to give an absolute measure of the optimisation abilities of each test algorithm.

4.4 The 2d Test Functions

To the best of the author’s knowledge, there are no published examples of test functions developed primarily to assess the performance of regional decomposition algorithms. Furthermore if one looks at the closest related field, multi-modal optimisation, the number of test functions available are very limited, Shekel’s Foxholes (section 7.4.1) and the modified Himmelblau function (section 2.2.3.1) being the most common examples. However both of these functions have been shown to be limited in both regularity (Shekel’s Foxholes) and complexity (Himmelblau) [Watson 1999]. The two test functions used here are more irregular and complex than the aforementioned examples and subsequently present more difficult decomposition tasks. The high performance regions within each test function are defined on an apriori basis, if a solution lies within a defined boundary, it is considered to be a high performance solution populating the given region. The first function (TF 1), introduced in section 2.2.3.3, consists of seven regions of high performance (figures 4.1 & 4.2).

![Figure 4.1: Test function 1 (TF1)](image)

![Figure 4.2: Test function 1 high performance regions](image)
A mathematical definition of test function 2 (TF 2) is given in appendix A. It consists of five regions of high performance (figures 4.3 & 4.4). Regions 2, 3 and 4 are of approximately similar robustness and magnitude. Region 5 is more isolated and sensitive than its neighbours. The multiple peaks contained within region 1 are highly sensitive with large changes in objective function fitness for small perturbations in the input values and represent the presence of noise within the search space.

\[ \text{Figure 4.3: Test function 2 (TF 2)} \]

\[ \text{Figure 4.4: Test function 2 high performance regions} \]

### 4.5 Results

Each test algorithm is run over 100 independent trials, after each trial performance metrics 1, 2, 3, 6 and 7 of table 4.1 are calculated. On completion, an average of metric 1 and 2 is taken over 100 trials and GDE and set cover standard deviation calculated.

The low GDE and FCS\text{avg} results for the DC algorithm indicate the most exploratative search of any of the test algorithms (tables 4.3 and 4.4). This is due to low convergence rates caused by random selection and replacement of the most similar. In both test cases, set cover is considerably larger than vmCOGA. The overall effect of this is that in seven of the 13 high performance regions DC outperforms vmCOGA in terms of regional decomposition (figures 4.5 and 4.6), and outperforms all remaining test algorithms in
regions 1 and 2 of test function 2. However, as suggested in sections 4.2.3 the increased set cover may be due to the increased reproductive activity. Furthermore, low GDE indicates many FCS solutions that lie outside of any high performance region; in TF 2 this figure approaches 50%. The low standard deviation of both GDE and set cover suggest reduced algorithm sensitivity, a possible reason for this being reduced replacement error [Mahfoud 1992]. Comparatively small FCSbest in both test functions is caused by a reduction is local exploitation caused by more explorative search.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Test Function 1</th>
<th>Test Function 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>vmCOGA</td>
<td>586.46 (7.84%)</td>
<td>560.35 (5.49%)</td>
</tr>
<tr>
<td>HUX / IP</td>
<td>540.71 (11.03%)</td>
<td>444.85 (9.15%)</td>
</tr>
<tr>
<td>DC</td>
<td>744.47 (7.48%)</td>
<td>653.09 (4.66%)</td>
</tr>
<tr>
<td>ECOGA (10x10)</td>
<td>220.42 (16.96%)</td>
<td>287.01 (15.73%)</td>
</tr>
<tr>
<td>ECOGA (20x20)</td>
<td>331.32 (5.68%)</td>
<td>361.45 (6.14%)</td>
</tr>
<tr>
<td>ECOGA (30x30)</td>
<td>830.65 (3.17%)</td>
<td>680.02 (2.40%)</td>
</tr>
</tbody>
</table>

Table 4.2: Set Cover and Set Cover Sensitivity
(values in parenthesis are standard deviations expressed as a % of measured value)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Test Function 1</th>
<th>Test Function 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>vmCOGA</td>
<td>71.78 (10.35%)</td>
<td>87.93 (3.95%)</td>
</tr>
<tr>
<td>HUX / IP</td>
<td>69.86 (17.50%)</td>
<td>91.87 (8.02%)</td>
</tr>
<tr>
<td>DC</td>
<td>57.74 (7.93%)</td>
<td>77.09 (4.60%)</td>
</tr>
<tr>
<td>ECOGA (10x10)</td>
<td>100.00 (0.00%)</td>
<td>99.16 (1.56%)</td>
</tr>
<tr>
<td>ECOGA (20x20)</td>
<td>100.00 (0.00%)</td>
<td>95.15 (2.27%)</td>
</tr>
<tr>
<td>ECOGA (30x30)</td>
<td>70.80 (4.47%)</td>
<td>88.02 (1.65%)</td>
</tr>
</tbody>
</table>

Table 4.3: GDE and GDE Sensitivity
(values in parenthesis are standard deviations expressed as a % of measured value)

The results from the HUX / IP algorithm are disappointing and offer little improvement upon vmCOGA, the regional decomposition of HUX / IP being outperformed by vmCOGA in the majority of high performance regions. Excluding the 10x10 ECOGA grid, HUX / IP also gives the most sensitive results. It should be stated that the optimal setting of the incest thresholds used during search may yield more promising results, but this requires an unacceptable level of apriori into the tuning process.
Premature convergence is noted in both the 10x10 and to a lesser degree the 20x20 ECOGA grids (very high GDE and FCSave). The subsequent stagnation of search also significantly reduces set cover in both algorithms. The very low GDE sensitivity in table 4.3 is misleading since it is caused by consistent premature convergence upon a limited number of regions (region 1 and 2 of TF 1 and regions 3 and 5 of TF 2). Set cover standard deviation gives a much clearer indication of sensitivity and indicates that the 10x10 ECOGA is the most susceptible to the effects of genetic drift. Relatively large FCSave and FCSbest values are unsurprising and may be attributed to the rapid convergence upon and subsequent exploitation of high performance regions.

![Figure 4.5: Regional decomposition for test function 1](image-url)
The premature convergence noted with the smaller ECOGA grids is eliminated in the 30x30 grid, but to achieve this a population size of 900 is required. The GDE and FCS results in both test cases indicate a similar degree of FCS convergence to that encountered with vmCOGA. However a significant increase in set cover produces the best regional decomposition results of any of the test algorithms. A further reduction in both GDE and set cover sensitivity suggests that the 30x30 ECOGA is generally the optimal algorithm across all the performance criteria.

![Figure 4.6: Regional decomposition results for test function 2](image)

4.6 The Six Dimension (6d) Test Function

The performance of the DC and ECOGA algorithms under higher dimensionality conditions is investigated by developing a multi-modal, high dimensionality test function and specifying the regions of high performance with the search space. However this is by no means a simple matter. In one or two dimensions, the search space may be easily visualised using techniques such as surface or contour plotting. However, as dimensionality is increased, the visualisation and subsequent interpretation of the test function becomes increasingly problematic. Exhaustive search offers a solution, but
exponential runtimes and the need to store each solution offline, quickly renders this technique impractical as dimensionality increases.

There are numerous test functions upon which one may assess GA performance under higher dimensionality. The DeJong test suite [DeJong 1975] consists of five test functions all possessing different characteristics. DeJong developed the suite to assess the performance of single optima seeking GAs, and as such is of limited use when assessing the performance of design space decomposition algorithms. More recently, specific multimodal test functions have been developed. Goldberg discusses a massively multi-modal 30 bit deceptive function [Goldberg et. al. 1992] where the fitness of a solution is dependent upon the unitation (number of ones) of the binary string. Although this function is extremely useful when assessing the performance of a multi-modal algorithm, the difficulty in defining regions of high performance within a binary search space reduces its utility. Turning to more practical design domains, the MDO test suite [Padula et. al. 1996] consists of a number of “real world” test functions developed to provide a standard set of test problems for multi-disciplinary design optimisation. Unfortunately only the location of the global optima are given, information relating to the local areas surrounding the optimal peaks is limited, this again renders the use of techniques such as exhaustive search necessary.

The test function presented here overcomes such problems whilst providing a more demanding decomposition task. This is achieved by using a six dimensional test function, consisting of three local, 2-dimensional planes denoted by the subscripts 1, 2 and 3 (figures 4.7, 4.9 and 4.11). The fitness $z$ of each sub plane is a function of two of the six input variables (a, b, c, d, e and f) where,
\[ z_1 = fn(a,b) \quad a,b \in [0,1] \]
\[ z_2 = fn(c,d) \quad c,d \in [0,1] \]
\[ z_3 = fn(e,f) \quad e,f \in [0,1] \]

\[ z_{1_{\text{max}}} = z_{2_{\text{max}}} = z_{3_{\text{max}}} = 0.0 \]
\[ z_{1_{\text{max}}} = z_{2_{\text{max}}} = z_{3_{\text{max}}} = 0.5 \]

If the total (global) fitness of a solution, \( f_{\text{global}} \), is the summation of the three planar fitnesses from each plane,

\[ f_{\text{global}} = z_1 + z_2 + z_3 \]

it follows,

\[ f_{\text{global}} = fn(a,b,c,d,e,f) \]
\[ f_{\text{global}_{\text{max}}} = 0 \quad f_{\text{global}_{\text{max}}} = 1.5 \]

In this sense each local plane acts as an individual level of the overall global test function therefore acting in a similar manner to Goldberg’s 30-bit function. This technique not only provides a more demanding optimisation task but also allows the easy declaration of all local and global high performance regions within the search space. When constructing the individual test function sub-planes several characteristics are evident,

- Peak robustness / sensitivity (Investigated in Plane 1)
- Multi-modality (Investigated in Plane 2)
- Deception (Investigated in Plane 2)
- Discontinuity (Investigated in Plane 3)
- Noise (Investigated in Planes 1 and 3)

The three local planes can be seen in figures 4.7 – 4.12, mathematical descriptions are given in appendices B, C and D. The first contains three peaks of varying sensitivity
Figure 4.7: Plane 1 (no added noise)

Figure 4.8: Plane 1 high performance regions

Figure 4.9: Plane 2

Figure 4.10: Plane 2 high performance regions

Figure 4.11: Plane 3 (no added noise)

Figure 4.12: Plane 3 high performance regions
(region 1i is the least sensitive and region 1iii is the most sensitive). The magnitude of each peak is 0.5, ±10% random noise is added to the function.

Plane two contains two diagonally opposed, highly multi-modal regions. Region 2i has twice as many optima, which are twice as sensitive as those contained within region 2ii. Plane two also contains two deceptive regions (located at the top right and bottom left of figure 4.10) that distract search away from regions 2i and 2ii. To maintain the modality of regions 2i and 2ii, random noise is not added to the plane.

The third plane contains three discontinuous plateau regions of high performance (3i, 3ii and 3iii), as with plane one, ±10% noise is added to this plane.

A solution is defined as being of global high performance if its fitness lies within 90% of the global optimal fitness, in this case 1.35. For a given solution, it follows that if each of its planar fitnesses exceed 0.45 the solution must lie within a global high performance region. However there are instances where a solution may possess a planar fitness of less than 0.45, but will have a global fitness of 1.35 or greater, a hypothetical example being $z_1 = 0.5, z_2 = 0.5, z_3 = 0.35$. Local high performance regions are therefore defined as areas where the planar fitness is greater than 0.35 (figures 4.8, 4.10 and 4.12). However, a further fitness check must be made since a proportion of solutions lying within each planar region may not be of global high performance, consider the example $z_1 = 0.35, z_2 = 0.35, z_3 = 0.35$. Consequently, for a solution to be a member of a global high performance region, it must first belong to a planar high performance region within all three planes and also possess a global fitness greater than 1.35. This defines 18 regions that are shown in table 4.5.
Table 4.5: The global high performance regions (HP = High performance)

4.7 Application of Test Algorithms to 6d Function

There is only one change made to the test algorithm of section 4.2 when moving from the two to six dimension application, this is an increase in the generation vector,

\[ g = \{25, 50, 75, 100, 125\} \]

When making a comparative study, the value of the filtering vector is arbitrary. However, to increase accuracy and interpretation of the results, \(R_f\) is adjusted until the GDE corresponding to \(\text{vmCOGA}\) (the control algorithm) lies approximately at the midpoint of its range (c50%). This allows for the maximum GDE deviation of the test algorithms in both the positive and negative directions, the corresponding filtering vector is found to be,

\[ R_f = \{1.75, 1.75, 1.75, 1.75, 1.75\} \]

Table 4.6: FCS results for DC and the 30x30 ECOGA
(values in parenthesis are standard deviations expressed as a % of measured value)

The performance characteristics of both the DC and ECOGA algorithms noted in the 2d study are again evident, and to a considerably greater extent when applied to the 6d study. The results relating to the DC algorithm (table 4.6) give further evidence of lower levels of convergence (very low GDE and \(\text{FCS}_{\text{ave}}\)). As in the 2d study, the set cover is greater than...
that noted with vmCOGA, however the regional decomposition results are poor. The low GDE responsible for this is also symptomatic of large proportions (c90%) of the FCS that lie outside of any high performance region. It is suggested that the lack of convergence within the DC algorithm increases global exploration and search space sampling at the expense of local exploitation. When compared with vmCOGA, such exploration increases the generation of low performance solutions, producing generally larger set cover but at a considerable decrease in GDE. Recent research has also identified this drop in DC performance upon application to more complex search domains [Watson 1999].

The complete opposite is noted in the results given by the 30x30 ECOGA, where rapid convergence produces a much smaller set cover and larger GDE when compared with vmCOGA. The resulting regional decomposition although offering a considerable improvement over DC, is still noticeably lower than vmCOGA (figure 4.13).

![Regional decomposition results for the 6d test function](image)

**Figure 4.13:** Regional decomposition results for the 6d test function

It is suggested that increasing the local and global ECOGA population size and / or increasing mutation may reduce search convergence to acceptable levels. However, doing
this reintroduces the need to set such parameters and makes no guarantee that such settings will give consistent results across any test function. Furthermore, higher mutation may induce the associated problems discussed in section 4.1. Finally the quadratic increase in population size and subsequent increase in computation load must also be taken into account when grid topology is increased.

4.8 Hybridising the ECOGA and DC Algorithms

The results from the 6d study suggest that there is little practicality in the generic application of either the DC or ECOGA algorithms. However this somewhat damning conclusion should not be made before the possible hybridisation of both algorithms is investigated. Hybridisation may increase GA performance by the integration of multiple algorithms into a single integrated search technique [Gen et. al. 1997, Parmee et. al. 1997b]. In this case search is split between both the ECOGA and DC. In using both algorithms during search it is suggested that a trade-off may be achieved, where one algorithm tempers to some degree the convergence rate of its contemporary. For example, an ECOGA / DC hybrid would firstly use ECOGA search to converge upon regions of high performance. At a predefined point, search switches to the DC algorithm, which reduces convergence and increases local set cover about each region. Nevertheless, there is a problem when hybridising the ECOGA and DC algorithms, this occurs when the population size is reduced or increased when search switches from one algorithm to the other.

It has been shown (section 2.1.1.5) that the ECOGA population evolves into species of solutions located at specific regions of the 2d grid. The position of the chromosomes upon the grid is therefore critical to the performance of the ECOGA. It is this topological speciation that is lost when search is switched from the DC to the ECOGA algorithms. Furthermore, an increase in population size requires the generation of a number of new
population members. Switching from the ECOGA to the DC search proves less problematic and requires only a reduction in population size. However solutions covering the whole grid should be copied to the DC population to ensure maximum niche sampling. Consequently, two mapping techniques are presented: ECOGA to DC and DC to ECOGA population mapping.

When switching from an ECOGA grid of population size $POPSIZE_{ECOGA}$ to a DC population of population size $POPSIZE_{DC}$, every $n$th member of the ECOGA population is copied to the DC population where,

$$n = \frac{POPSIZE_{ECOGA}}{POPSIZE_{DC}}$$ ...

DC to ECOGA population mapping uses a method similar to restarts [Eshelman 1991]. In this example every DC population member is used as a template, $m$ new ECOGA population members are then generated by randomly selecting and flipping a fixed proportion (0.035) of the template bits where,

$$m = \frac{POPSIZE_{ECOGA}}{POPSIZE_{DC}} - 1$$ ...

As one would expect, as the proportion of search given to the ECOGA in both hybrid algorithms increases, the degree of convergence within in the FCS increases (table 4.7 and 4.8).

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Set cover</th>
<th>GDE</th>
<th>FCS$_{avg}$</th>
<th>FCS$_{best}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>vcCOGA</td>
<td>285.72 (28.00%)</td>
<td>42.11% (30.16%)</td>
<td>1.345</td>
<td>1.471</td>
</tr>
<tr>
<td>DC (5:0)</td>
<td>306.54 (20.78%)</td>
<td>10.75% (35.73%)</td>
<td>1.285</td>
<td>1.452</td>
</tr>
<tr>
<td>4:1</td>
<td>303.00 (17.19%)</td>
<td>12.57% (29.90%)</td>
<td>1.290</td>
<td>1.458</td>
</tr>
<tr>
<td>3:2</td>
<td>252.41 (18.26%)</td>
<td>18.94% (23.18%)</td>
<td>1.297</td>
<td>1.461</td>
</tr>
<tr>
<td>2:3</td>
<td>195.70 (20.17%)</td>
<td>30.98% (22.60%)</td>
<td>1.315</td>
<td>1.467</td>
</tr>
<tr>
<td>1:4</td>
<td>140.89 (10.11%)</td>
<td>52.59% (12.13%)</td>
<td>1.348</td>
<td>1.473</td>
</tr>
<tr>
<td>ECOGA (0:5)</td>
<td>101.22 (12.40%)</td>
<td>95.33% (4.60%)</td>
<td>1.410</td>
<td>1.475</td>
</tr>
</tbody>
</table>

Table 4.7: FCS results for the DC / ECOGA hybrid (the ratio in the first column relates to the proportion of search given to the respective algorithm, values in parenthesis are standard deviations expressed as a % of measured value
Algorithm | Set cover | GDE | FCS_{ave} | FCS_{best}
--- | --- | --- | --- | ---
vmCOGA | 285.72 (28.00%) | 42.11% (30.16%) | 1.345 | 1.471
ECOGA (5:0) | 101.22 (12.4%) | 95.33% (4.60%) | 1.410 | 1.475
4:1 | 91.49 (15.84%) | 95.09% (4.46%) | 1.408 | 1.475
3:2 | 114.60 (22.46%) | 69.92% (19.01%) | 1.375 | 1.471
2:3 | 178.78 (22.28%) | 40.42% (22.60%) | 1.335 | 1.466
1:4 | 237.48 (26.20%) | 22.29% (31.22%) | 1.307 | 1.461
DC (0:5) | 306.54 (20.78%) | 10.75% (35.73%) | 1.285 | 1.452

Table 4.8: FCS results for the ECOGA / DC hybrid (the ratio in the first column relates to the proportion of search given to the respective algorithm, values in parenthesis are standard deviations expressed as a % of measured value.

Conversely, the degree of convergence within the FCS decreases as the proportion of DC search increases. The configurations that give the closest degree of FCS convergence (determined by GDE and FCS_{ave}) when compared with the vmCOGA control algorithm are the 1:4 DC / ECO and 2:3 ECOGA / DC hybrids. In each case the standard deviation of both GDE and set cover are far smaller than the corresponding vmCOGA values. However notably lower set cover in both hybrids result in regional decomposition results that are considerably less than vmCOGA and although they outperform pure DC search they offer little improvement upon the ECOGA algorithm (figure 4.14).

Considerable empirical effort has focussed upon developing an optimal hybrid configuration that maintains a degree of FCS convergence equal to or higher than vmCOGA whilst eliminating the characteristic drop in set cover noted previously. The most promising hybrid found is the 1:3:1 DC / ECOGA / DC hybrid. In this example slow initial convergence allows for global exploration of the search space. As search switches to the ECOGA, the evolving population converges upon high performance regions. Upon returning to DC search, the convergence rate reduces and the populated regions are explored locally by the DC algorithm. The regional decomposition results for the 1:3:1 DC / ECOGA / DC algorithm show some degree of promise (figure 4.14). In 12 of the 18 regions the hybrid outperforms the pure ECOGA algorithm and in seven of the 18 regions vmCOGA is outperformed.
However, when one refers to the remaining performance measures (table 4.9), it can be seen that the degree of convergence within the FCS is considerably lower than vmCOGA. The increase in set cover is sufficiently large to overcome the lower GDE and produce the improvement over vmCOGA in the seven regions. As in previous examples, the low GDE indicates large proportions of the FCS the lie outside of all the defined high performance regions. Furthermore the 1:3:1 DC / ECOGA / DC hybrid is generally more sensitive than the DC, ECOGA, vmCOGA and the hybrid algorithms of figure 4.

Table 4.9: FCS results for the 1:3:1 DC/ECOGA/DC hybrid (the ratio in the first column relates to the proportion of search given to the respective algorithm, values in parenthesis are standard deviations expressed as a % of measured value

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Set cover</th>
<th>GDE</th>
<th>FCS_{best}</th>
<th>FCS_{avg}</th>
</tr>
</thead>
<tbody>
<tr>
<td>vmCOGA</td>
<td>285.72 (28.00%)</td>
<td>42.11% (30.16%)</td>
<td>1.345</td>
<td>1.471</td>
</tr>
<tr>
<td>DC</td>
<td>306.54 (20.78%)</td>
<td>10.75% (35.73%)</td>
<td>1.285</td>
<td>1.452</td>
</tr>
<tr>
<td>ECOGA (30x30)</td>
<td>101.22 (12.40%)</td>
<td>95.33% (4.60%)</td>
<td>1.410</td>
<td>1.475</td>
</tr>
<tr>
<td>DC / ECOGA / DC</td>
<td>379.10 (26.70%)</td>
<td>26.79% (32.13%)</td>
<td>1.314</td>
<td>1.471</td>
</tr>
</tbody>
</table>

4.9 Concluding Remarks

The results of this study clearly illustrate the difficulties in selecting a diverse search engine that performs sufficiently well across a number of domains. If the 2d results were
taken in isolation, the 30x30 ECOGA would appear to be the superior algorithm in both robustness and also overall decomposition ability. However, as dimensionality increases, convergence during ECOGA search becomes too great and global exploration is lost. Conversely, DC gives much slower rates of convergence which produce large set cover but at the expense of a reduction in GDE, these characteristics are again augmented by application to a higher dimensionality test function. It is suggested that increasing the population size and / or mutation probability during search may reduce the convergence rate of the ECOGA, however the subsequent quadratic increase in population size (due to grid topology) and the need to set appropriate mutation rates makes both approaches infeasible.

A degree of success has been achieved by hybridising both algorithms, where the contrasting convergence rates are tempered by splitting search between both algorithms. However the increase in performance is marginal and the protracted empirical investigation needed to achieve this provides very little justification for any further investigation of the DC, ECOGA or their hybridised algorithms.
5.1 Introduction

One of the major drawbacks of both the DC and 30x30 ECOGA algorithms presented in chapter four is the inability to adequately specify and control the convergence rate before and during search. Hence the success of the algorithms when applied to both TF1 and TF2 is lost when applied to the higher dimensionality 6d function.

In the case of highly convergent search, the population rapidly converges upon a handful of peaks (the 10x10 and 20x20 ECOGA results being a prime example). The ability of a niche or peak to sustain solutions within the population is dependent upon both the maximum fitness and the local fitness landscape of the niche. If the optima is highly sensitive (large changes in fitness for small input perturbations) it may only be able to sustain a handful of solutions, constituting only a fraction of the overall population. As search continues, solutions from other, more robust regions quickly dominate the evolutionary process. The subsequent elimination of the sensitive peak from the evolving population is probable.

In addition to this it is unlikely that a simple GA will be able to relocate the eliminated region [Cobb et. al. 1995]. When seeking a single region or optima this is of lesser importance. Nevertheless, in terms of identifying multiple high performance regions within the an engineering environment it is more critical, since a more sensitive peak may
offer considerable utility when further performance criteria such as cost of materials or ease of manufacture are considered\(^1\).

The elimination of solutions located upon sensitive peaks within the population may be curtailed (but not eradicated completely) by reducing the convergence rate and / or increasing the sampling of the search space. If however, the convergence rate is too low (as is the case with the DC algorithm) large numbers of low fitness solutions that do not lie within high performance regions will pass to the FCS. Furthermore, reducing the convergence rate alone does not necessarily increase the possibility of locating novel regions. Increasing the sampling of the search space offers a more appealing solution since this not only reduces the convergence rate but also increases the degree of global exploration and therefore the possibility of locating unidentified or previously eliminated peaks.

Consequently, the goal of the research described in this chapter is the development of search algorithms that are able to efficiently sample the search space, increasing the probability of locating novel regions and reducing the possibility of their elimination during search. Furthermore the degree of sampling should not be static since exploration is sacrificed for exploitation during the later stages of search. Two approaches that increase search space sampling are presented. Halton injection COGA (HiCOGA) and Spatial selection COGA (SsCOGA). In HiCOGA, sampling is explicitly achieved by injecting diverse individuals directly into the reproductive phase. With SsCOGA, sampling is implied by utilising a dual selection scheme where it is more probable that isolated individuals are selected for reproduction.

\(^1\) It is acknowledged that when compared with regions of greater robustness there is an increased likelihood that a sensitive peak may be a result of the coarseness of the conceptual design model, and should therefore be treated with a degree of caution. Nevertheless, its presence should be indicated to the designer.
5.2 Halton Injection COGA (HiCOGA)

The injection of randomly generated strings directly into the evolutionary process is a characteristic of the CHC algorithm where search is reinitialised when stagnation occurs (section 4.2.2). Grefenstette [Grefenstette 1992] supplements a reduced mutation rate by replacing a percentage of the population at each generation by randomly generated chromosomes, called random immigrants. This ensures that a level of sampling and exploration is maintained throughout the search space. More recently, Lin et. al. [Lin et. al. 1994] identifies a drawback with such techniques called the non-effect problem, this occurs when the injected chromosomes are not selected for crossover and therefore do not participate in the evolutionary process, this increases if the relative fitness of the injected chromosomes is poor. In addition to this, it has been questioned whether the random generation of chromosomes is the most efficient means of ensuring optimal sampling of the search space [Kallel et. al. 1997].

HiCOGA addresses these problems by eliminating mutation and injecting solutions generated from a low discrepancy sequence (section 5.2.1) directly into the crossover phase. Using low discrepancy points ensures more efficient search space sampling when compared with randomly generated points, whilst injecting directly into the crossover phase overcomes the non-effect problem.

The mechanism of HiCOGA is shown in figure 5.1. Solutions selected using objective function fitness are paired and reproduce with injected chromosomes. As the number of injected chromosomes increases, a greater proportion of diverse individuals take part in crossover, the sampling of the search space increases and the convergence rate slows. It is suggested that the number of injections made per generation may be used to attain a balance between exploration and exploitation (in the same way as mutation is employed in vmCOGA). During the initial stages of search high numbers of injections ensures
maximal sampling of the search space. As search progresses the number is relaxed to promote the convergence upon high performance regions of the search space.

**Figure 5.1**: The reproduction phase in HiCOGA

<table>
<thead>
<tr>
<th>If $n_Hi \leq \frac{n_{max}}{2}$</th>
<th>If $n_Hi &gt; \frac{n_{max}}{2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select $n_{Hi}$ individuals using FPS. Reproduce with LD individuals to produce $2 \times n_{Hi}$ children.</td>
<td>Select $(n_{cross} - n_{Hi})$ individuals using FPS. Reproduce with $(n_{cross} - n_{Hi})$ LD individuals to produce $2 \times (n_{cross} - n_{Hi})$ children</td>
</tr>
<tr>
<td>Select the remaining $(n_{cross} - 2n_{Hi})$ parents using FPS. Reproduce to produce the remaining $(n_{cross} - 2n_{Hi})$ children</td>
<td>Randomly pair and reproduce the remaining $n_{cross} - 2(n_{cross} - n_{Hi})$ LD individuals to produce remaining $n_{cross} - 2(n_{cross} - n_{Hi})$</td>
</tr>
<tr>
<td>Randomly replace $n_{cross}$ children into population</td>
<td>Randomly replace $n_{cross}$ children into population</td>
</tr>
</tbody>
</table>

**5.2.1 Low Discrepancy and the Leaped Halton Sequences**

Low Discrepancy (LD) or quasirandom sequences generate any number ($m$) of points that uniformly fill an $n$-dimensional unit hypercube. LD sequences are commonly applied in techniques such as Monte Carlo integration where a diverse sampling of the search space is important [Bratley et. al. 1992, Athan et. al. 1996].

As the name suggests, quasi-random sequences are not stochastic in the strictest sense. The generated points are maximally different, hence the sequences may be more accurately considered as deterministic in nature. One of the most common approaches is the Sobol sequence [Sobol 1967]. In this example, each LD sequence is generated by a complex technique requiring the use of both primitive polynomials and a set of direction numbers (a
comprehensive review of the Sobol technique is given by Bratley et. al. [Bratley et. al. 1988]). Tables of primitive polynomials are widely available [Lin et. al. 1983]. However, the list of direction numbers suggested by Sobol [Sobol 1976] is limited to only the first few dimensions. This drawback coupled with the mathematical complexity of the Sobol sequence limits its application to higher dimensionality sequences [Bratley et. al. 1988]. The Halton sequence [Halton 1960] offers a far simpler approach by manipulating binary fractions. A Halton sequence of \( m \) points within an \( n \)-dimensional hypercube may be generated as follows,

1. Express \( i \) in base \( j \),

   Where \( i = 1, 2, 3, \ldots, m \) and \( j \) is the \( n \)th prime number

If the 7th Halton number within the 3rd dimension is to be found,

6 expressed in base 5 (3rd prime number) is 12 (base 5)

2. Reverse the order of \( i \),

   12 (base 5) \( \Rightarrow \) 21 (base 5)

3. Place a radix point in front of \( i \),

   21 (base 5) \( \Rightarrow \) 0.21 (base 5)

4. Determine the value of the corresponding modulo fraction,

\[
0.21 \text{ (base 5)} = \frac{11}{5^2} = 0.44
\]

<table>
<thead>
<tr>
<th>( i )</th>
<th>1st Dimension (base 2)</th>
<th>2nd Dimension (base 3)</th>
<th>3rd Dimension (base 5)</th>
<th>4th Dimension (base 7)</th>
<th>5th Dimension (base 9)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.50</td>
<td>0.33</td>
<td>0.20</td>
<td>0.14</td>
<td>0.11</td>
</tr>
<tr>
<td>2</td>
<td>0.25</td>
<td>0.66</td>
<td>0.40</td>
<td>0.29</td>
<td>0.22</td>
</tr>
<tr>
<td>3</td>
<td>0.75</td>
<td>0.11</td>
<td>0.60</td>
<td>0.43</td>
<td>0.33</td>
</tr>
<tr>
<td>4</td>
<td>0.13</td>
<td>0.44</td>
<td>0.80</td>
<td>0.57</td>
<td>0.44</td>
</tr>
<tr>
<td>5</td>
<td>0.63</td>
<td>0.77</td>
<td>0.04</td>
<td>0.71</td>
<td>0.55</td>
</tr>
<tr>
<td>6</td>
<td>0.38</td>
<td>0.22</td>
<td>0.24</td>
<td>0.85</td>
<td>0.66</td>
</tr>
<tr>
<td>7</td>
<td>0.88</td>
<td>0.55</td>
<td>0.44</td>
<td>0.02</td>
<td>0.77</td>
</tr>
<tr>
<td>8</td>
<td>0.07</td>
<td>0.88</td>
<td>0.64</td>
<td>0.16</td>
<td>0.88</td>
</tr>
<tr>
<td>9</td>
<td>0.56</td>
<td>0.04</td>
<td>0.84</td>
<td>0.31</td>
<td>0.01</td>
</tr>
<tr>
<td>10</td>
<td>0.31</td>
<td>0.37</td>
<td>0.08</td>
<td>0.45</td>
<td>0.12</td>
</tr>
</tbody>
</table>

Table 5.1: A sample Halton sequence of ten points in five dimensions (\( m = 10, n = 5 \)
Halton numbers generated within the first five dimensions are given in table 5.1. Apart from its overall simplicity, the main advantage of the Halton sequence lies in the elimination of primitive polynomial and direction number selection. However, the Halton sequence does suffer from a significant drawback, this being a disintegration of quasi randomness of the sequence for \( n > 16 \). The Leaped Halton (LH) sequence [Kocis et al. 1997] overcomes this by using only every \( l \)th value of \( i \) to generate the sequences (\( l \) is a prime number that differs from all bases used, in this instance the \((n+1)th\) prime is used). Using this approach, the Leaped Halton sequence may be used to generate any number of quasi-random points within a hypercube of unbounded dimensionality. Moreover the generated points may be mapped to an orthogonal search space of any desired size and scale.

### 5.2.2 Transforming the Leaped Halton Sequence

It is apparent that the lack of intrinsic randomness in a leaped Halton sequence results in every generated array of \( m \) points in \( n \) dimensional space being identical. A degree of randomness must therefore be induced into each subsequent \( n \times m \) sequence without disrupting the overall low discrepancy of the points. This is achieved by adding a randomly generated factor to transform the generated sequence.

![Figure 5.2: Transforming the LH sequence](image)
Consider an $n$-dimensional point, generated using the LH sequence and presented in a 2-dimensional unit hypercube, $i,j$ (the edges of the hypercube are wrapped to form a toroidal topology) (figure 5.2). If a random number of the interval $[0,1]$ is generated and then added to every dimensional component of the $m$ points ($\mathbf{a}$ and $\mathbf{f}$ in this instance), the original point is transformed to a new point within the hypercube.

If the first 60 points of a LH sequence are plotted in a hyperplane, the maximally different nature of the points is clearly evident (figure 5.3). If the sequence is then transformed it can be seen that the centroid of original sequence is projected to a new location (figure 5.4), but the relative position and the subsequent low discrepancy of the sequences remain unaffected.

![Figure 5.3: A 60 point LH sequence represented in a 2-dimensional hyperplane (The centroid of the hyperplane is indicated by the chequered circle)](image1)

![Figure 5.4: A transformed 60 point LH sequence represented in a 2-dimensional hyperplane (The transformed centroid of the hyperplane is indicated by the chequered circle)](image2)

To compare the degree of sampling within a leaped Halton and random sequence, an $n \times m$ ($\text{number of variables} \times \text{population size}$) transformed LH sequence is compared with an $n \times m$ sequence generated randomly. The average nearest neighbour Euclidean distance of the points within each sample gives a measure of the dispersion of the sequence and is used to indicate the degree of sampling (the metric increases in proportion to the degree of
sampling). The results are averaged over 100 independent trials and in each case a different seed is used to initialise the random number.

As expected, there is a general reduction in the dispersion metric as population size increases, caused by an increase in the solution density of the sample space (figure 5.5). Conversely, as the number of dimensions (variables) increases the dispersion metric increases, caused by the expansion of the search space (figure 5.6). In both plots the dispersion metric and the subsequent sampling of the transformed LH sequence is noticeably higher than that achieved with random points, indicating considerable improvement in sampling when using the transformed LH sequence.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure5_5}
\caption{Effect of increasing population size $m (n = 2)$}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure5_6}
\caption{Effect of increasing dimensionality $n (m = 50)$}
\end{figure}

5.3 Spatial Selection COGA (SsCOGA)

Unlike HiCOGA, which replaces mutation by injecting diverse chromosomes, spatial selection is used in conjunction with a mutation rate of 0.001 [DeJong 1975]. A dual selection scheme promotes the selection of individuals within isolated regions of the search space. Upon initial inspection it may seem that this philosophy is similar to that which inspired fitness sharing, however closer investigation reveals that this is not the case. Fitness sharing seeks to reduce the overpopulation and subsequent over exploration of
regions of high performance by reducing the chance that such solutions are selected for reproduction. Sharing therefore does not necessarily imply an increased sampling of the overall search space. SsCOGA promotes the selection of diverse individuals and therefore does increase the degree of sampling. Furthermore, the dual selection scheme of SsCOGA does not reduce the objective function fitness of an individual and therefore does not require the definition of complex functions, as is the case with sharing. As its name suggests, dual selection selects parents in terms of two independent fitness criteria. The first parent is selected in terms of objective function fitness whilst the second is selected using a local solution density metric. The density metric ensures that a solution in a sparsely populated region is more likely to be selected (figure 5.7).

![Diagram of the reproduction phase in SsCOGA](image)

**Figure 5.7:** The reproduction phase in SsCOGA

- \( n_{ss} \) = number of spatial selections / generation
- \( n_{cross} \) = number of reproduced individuals / generation

As with HiCOGA, as more spatial selections are made, search becomes more explorative and the convergence rate decreases. Two local solution density metrics are presented, hypersphere density and nearest neighbour Euclidean distance.
5.3.1 Hypersphere Density Metric

The hypersphere density metric is the more intuitive approach and gives an absolute measure of solution density. If the theoretical population of figure 5.8 is enlarged to show the lower left quarter containing the most sensitive and isolated peak, it can be seen that for a given parent \((P_1)\) a hypersphere of radius \(r\), with centre at \(P_1\) is constructed. The local density count of \(P_1\) is the total number of individuals (including \(P_1\)) that lie within the hypersphere (figure 5.9.). The local density fitness of \(P_1\) is the inverse of the density count. It follows that as the density count, and the subsequent sampling in the local regions of \(P_1\) increases its local density fitness decreases.

There are two drawbacks associated with this technique. The first is the need to map the population to the interval \([0,1]\) to maintain consistency when constructing the hypersphere. The second and more problematic is the need to select the radius of the hypersphere, \(r\). This is a similar problem to selecting \(\sigma_{share}\) in Goldberg's sharing. To reduce the complexity of selecting \(r\), the following method is proposed. If an \(n\) dimensional unit search space is considered, \(r\) may be obtained by using simple Pythagorean analysis,
\[ r = \sqrt{\sum_{i=1}^{n} r_i^2} \quad r_i \in [0,1] \quad \ldots 5-1 \]

where \( r_i \) = hypersphere radius in each unit plane. To further simplify both the calculation of \( r \) and the interpretation of the corresponding results, the unity of the search space may be exploited so that,

\[ r_1 = r_2 = \ldots = r_n \quad \ldots 5-2 \]

Equation 5-1 reduces to,

\[ r = \sqrt{n r_{h1}^2} \quad r_{h1} \in [0,1] \quad \ldots 5-3 \]

where \( r_{h1} \) is the planar hypersphere radius.

### 5.3.2 Nearest Neighbour Euclidean Distance Metric

The second metric is more pragmatic since it eliminates the need to set \( r \). The inspiration for the nearest neighbour Euclidean measure comes from the realisation that at initialisation the separation of the population is generally at its largest (more so if a LD sequence is used for initialisation). As search continues and convergence begins, the general separation of the population decreases [Shine et. al. 1997]. Conversely, the distance between solutions in more underpopulated regions of the search remains comparatively large. Search space sampling is therefore increased by giving greater potential for selection to the more isolated individuals. For the same given parent \( (P_i) \) of figure 5.9 its local density fitness is the Euclidean distance between itself and its nearest neighbour (figure 5.10). As with the hypersphere metric, a disadvantage of this technique is the need to map the search space to the interval \([0,1]\). However it may be seen that solutions from less densely populated regions (upper left and lower left corners of figure 5.8), will have an increased probability of being selected when compared with the solutions in more densely populated regions (right hand side of figure 5.8).
5.4 Application of HiCOGA and SsCOGA to 6d Test Function

To assess the effect that the number of Halton injections and spatial selections has upon the performance of HiCOGA and SsCOGA respectively, a series of empirical investigations is made upon the 6d-test function detailed in section 4.6. In a similar manner to the mutation vector of vmCOGA, the Halton injection and spatial selection vectors define the number of Halton injections and spatial selections made during each search stage.

\[
Hi^T = \begin{bmatrix}
\text{Halton injections during search stage 1} \\
\text{Halton injections during search stage 2} \\
\text{Halton injections during search stage 3} \\
\vdots \\
\text{Halton injections during search stage n}
\end{bmatrix}
\]

\[
ss^T = \begin{bmatrix}
\text{spatial selections during search stage 1} \\
\text{spatial selections during search stage 2} \\
\text{spatial selections during search stage 3} \\
\vdots \\
\text{spatial selections during search stage n}
\end{bmatrix}
\]

The default search parameters and GA operators are consistent with those detailed in section 4.7.

Figure 5.10: The nearest neighbour Euclidean distance local density measure
5.4.1 Presentation of Results

When interpreting the results of algorithms such as SsCOGA and HiCOGA (designed primarily to increase the sampling of underpopulated high performance areas of the search space) it is useful to specify benchmark values for the comparable number of solutions located within each region.

As with all investigations throughout this thesis, vmCOGA is used as the control or benchmark algorithm. Figure 5.11 illustrates the number of FCS solutions lying within each of the 18 high performance regions; as taken from the vmCOGA run of section 4.7. The large differences in the number of FCS solutions within each high performance region are clearly evident. Furthermore, if the regions are ranked in ascending order of RD, a measure of the sensitivity of each region is indicated (table 5.2). Here a rank of one indicates the most sensitive region, whilst a rank of 18 indicates the least sensitive and most robust region.

![Regional sensitivity classification (vmCOGA)](image-url)
The results in the following section are presented in figures where regional sensitivity ranking is denoted upon the abscissa. These regions are also referred to as belonging to either high (rank 1-6), medium (rank 7-12) or low (rank 13-18) sensitivity groups. This not only simplifies the presentation of the results, but also gives a clearer indication of the effects of regional sensitivity upon the performance of each of the test algorithms.

<table>
<thead>
<tr>
<th>Regional sensitivity rank</th>
<th>Region</th>
<th>Average number of solutions within region (vmCOGA)</th>
<th>Sensitivity classification group</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>17</td>
<td>0.78</td>
<td>High</td>
</tr>
<tr>
<td>2</td>
<td>14</td>
<td>0.92</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>13</td>
<td>1.67</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>1.89</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>11</td>
<td>1.90</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>2.22</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>15</td>
<td>3.08</td>
<td>Medium</td>
</tr>
<tr>
<td>8</td>
<td>18</td>
<td>3.54</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>4.13</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>4.36</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>7</td>
<td>4.47</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>10</td>
<td>5.27</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>12</td>
<td>8.68</td>
<td>Low</td>
</tr>
<tr>
<td>14</td>
<td>9</td>
<td>9.76</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>4</td>
<td>10.96</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>1</td>
<td>11.26</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>6</td>
<td>19.33</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>3</td>
<td>20.91</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.2: Regional sensitivity rank

5.4.2 Results

The seven test algorithms are shown in table 5.3. It should be noted that throughout the course of this investigation, the Halton injection and spatial selection vectors are chosen to illustrate the specific operational characteristics of each algorithm and not to promote optimal performance. Throughout this study, the generation and filtering vectors concur with those of section 4.7.

<table>
<thead>
<tr>
<th>Figure</th>
<th>Algorithm</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.12</td>
<td>SsCOGA$_1$</td>
<td>(Euclidean density) ss = [25, 20, 15, 10, 5]</td>
</tr>
<tr>
<td>5.13</td>
<td>HiCOGA$_1$</td>
<td>$Hi = [25, 20, 15, 10, 5]$</td>
</tr>
<tr>
<td>5.14</td>
<td>SsCOGA$<em>3$ (Hypersphere density) ss = (20, 15, 10, 5, 0) $r</em>{st} = 0.35$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SsCOGA$<em>4$ (Hypersphere density) ss = (20, 15, 10, 5, 0) $r</em>{st} = 0.25$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SsCOGA$<em>5$ (Hypersphere density) ss = (20, 15, 10, 5, 0) $r</em>{st} = 0.15$</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.3: The test algorithms
If the SsCOGA and HiCOGA algorithms of table 5.4 are examined initially, it is clearly apparent that as the number of spatial selections and Halton injections decrease (moving from example one to two in each case), the degree of convergence within the FCS increases (increased GDE, FCSave and FCSbest). Furthermore the larger GDE, FCSave and FCSbest values in both SsCOGA examples when compared with their HiCOGA contemporaries suggests that the injection of Halton chromosomes has a greater effect upon the degree of convergence within the FCS. A further characteristic of table 5.4 is an increase in the standard deviation of both set cover and GDE when compared with vmCOGA, the exception to this being SsCOGA2 and here the difference in minimal. A possible explanation for this is that increasing the exploration of underpopulated regions of the search space may have the drawback of promoting over zealous exploration of areas that do not lie within any of the defined high performance regions. This influences the variation of GDE between trials and may increase its standard deviation. The further increase in standard deviation when the exploration of underpopulated regions is increased (SsCOGA1 and HiCOGA1 compared with SsCOGA2 and HiCOGA2) appears to support this argument. This effect, albeit in the opposite sense, has already been noted in the 10x10 and 20x20 ECOGA algorithms of chapter 4. Here premature convergence severely limits the exploration of the search space outside all but a handful of the most robust regions of high performance. Hence, over any number of independent trials search will continuously and rapidly converge upon such optima producing very low GDE standard deviation.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Set cover</th>
<th>GDE</th>
<th>FCSave</th>
<th>FCSbest</th>
</tr>
</thead>
<tbody>
<tr>
<td>vmCOGA</td>
<td>285.72</td>
<td>42.11%</td>
<td>1.344</td>
<td>1.471</td>
</tr>
<tr>
<td>SsCOGA1</td>
<td>236.61</td>
<td>55.59%</td>
<td>1.358</td>
<td>1.470</td>
</tr>
<tr>
<td>SsCOGA2</td>
<td>231.05</td>
<td>66.30%</td>
<td>1.371</td>
<td>1.473</td>
</tr>
<tr>
<td>HiCOGA1</td>
<td>202.29</td>
<td>33.28%</td>
<td>1.331</td>
<td>1.465</td>
</tr>
<tr>
<td>HiCOGA2</td>
<td>218.12</td>
<td>46.67%</td>
<td>1.350</td>
<td>1.470</td>
</tr>
</tbody>
</table>

Table 5.4: FCS results for SsCOGA1,2 and HiCOGA1,2 (values in parenthesis are standard deviations expressed as a % of measured value)
The effect of \textit{Hi} and \textit{ss} upon the sampling of the search space and the exploration of sensitive high performance regions is evident in the regional decomposition plots of figures 5.12 and 5.13. In four of the six most sensitive regions for SsCOGA and five of the six regions for HiCOGA, RD increases with an increase in the number of spatial selections and Halton injections. Conversely, in all of the least sensitive regions for both SsCOGA and HiCOGA, the algorithm with fewer selections or injections gives the superior performance.

Figure 5.12: Relative regional decomposition results for SsCOGA\textsubscript{1} and SsCOGA\textsubscript{2}

Figure 5.13: Relative regional decomposition results for HiCOGA\textsubscript{1} and HiCOGA\textsubscript{2}
In general, the lower RD results encountered with HiCOGA is a combination of both lower set cover and GDE. This suggests that the explicit sampling action of HiCOGA, although successful in promoting an increased exploration of sensitive regions, may be overly disruptive and therefore less useful than the implied sampling of SsCOGA. Nonetheless, the empirical evidence presented for both SsCOGA and HiCOGA appears to support the hypothesis that increasing the number of spatial selections or Halton injections not only reduces the degree of convergence within the FCS but also increases the sampling and subsequent exploration of the more sensitive regions.

When comparing the results of table 5.5 with SsCOGA₂, it may be seen that the replacement of the Euclidean local density measure with the hypersphere measure further reduces the degree of FCS convergence. However the value of \( r_{hi} \) itself appears to have little effect upon the convergence within the FCS.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Set cover</th>
<th>GDE</th>
<th>FCS_{avg}</th>
<th>FCS_{best}</th>
</tr>
</thead>
<tbody>
<tr>
<td>vmCOGA</td>
<td>285.72 (28.00%)</td>
<td>42.11% (30.16%)</td>
<td>1.344</td>
<td>1.471</td>
</tr>
<tr>
<td>SsCOGA₃</td>
<td>232.24 (33.45%)</td>
<td>59.02% (30.40%)</td>
<td>1.363</td>
<td>1.473</td>
</tr>
<tr>
<td>SsCOGA₄</td>
<td>251.97 (32.46%)</td>
<td>56.22% (30.61%)</td>
<td>1.360</td>
<td>1.472</td>
</tr>
<tr>
<td>SsCOGA₅</td>
<td>235.75 (29.32%)</td>
<td>59.84% (29.03%)</td>
<td>1.365</td>
<td>1.473</td>
</tr>
</tbody>
</table>

Table 5.5: FCS results for SsCOGA₃,₄ and ₅
(values in parenthesis are standard deviations expressed as a % of measured value)

Referring to figure 5.14, in ten of the twelve high and medium sensitivity regions, SsCOGA₄ (\( r_{hi} = 0.25 \)) provides the more superior results, due in part to a relative increase in set cover. However in the least sensitive regions, the relative performance of SsCOGA₄ diminishes substantially, and is superseded by both SsCOGA₁ and SsCOGA₅. Of these two algorithms, SsCOGA₅ using the smallest hypersphere radius outperforms its contemporaries in three of the six regions.
When comparing the hypersphere with the equivalent Euclidean measure SsCOGA algorithm (SsCOGA₂) there is little evidence to suggest any tangible benefits from using the hypersphere measure. When one firstly considers that the value of \( r_{h1} \) itself has been shown to affect the RD results in differing sensitivity groups and secondly that the setting of the \( r_{h1} \) parameter is highly subjective and introduces the need for apriori relating to the location and appearance of the peaks; the utility of the hypersphere measure diminishes considerably.

### 5.5 Concluding Remarks

This chapter has presented two novel COGA techniques, HiCOGA and SsCOGA, both have been designed to increase the sampling of the search space and promote increased exploration of the more sensitive regions contained within it. For each presented approach it has been shown that both the degree of FCS convergence and the RD of the high and low sensitivity regions may be modified by altering either the number of Halton injections or spatial selections during search. Of the two presented SsCOGA local solution density metrics, there is little empirical evidence to suggest the overall superiority of either.
However, the absence of the requirement to set parameters before execution when using the Euclidean metric makes it the more pragmatic approach, as a result the investigation of the hypersphere measure is discontinued here.

One of the most striking characteristics noted in this investigation is the large differences in the degree of FCS converge for the same Halton injection and spatial selection vectors. Although of lesser importance in the context of this chapter, this lack of continuity makes accurate comparison and practical application of either technique problematic. However, it is suggested that further empirical effort may determine values for $Hi$ and $ss$ may yield closer similarity in the degree of FCS convergence. Unfortunately this reintroduces the need to tune the algorithm parameters. Not only can this process be cumbersome and extremely protracted (the ECOGA / DC hybridisation investigation of chapter four being an example) but it also places considerable emphasis upon the user, and is therefore impractical.

As a result, the following chapter presents a technique that controls the number of Halton injections and spatial selections throughout search by using convergence data extracted from the evolving population. Not only does this eliminate the need to set the values for $Hi$ and $ss$ before execution, but also promotes consistent convergence rates during search and therefore similar levels of convergence within the FCS. This subsequently allows for a more exacting performance comparison of HiCOGA and SsCOGA with vmCOGA.
6.1 Introduction

The ability of HiCOGA and SsCOGA to increase the sampling of the search space and to promote exploration of the more sensitive regions contained within it, has been demonstrated in chapter 6. However, both approaches have only been used in an open loop sense (figure 6.1), where the number of Halton injections and spatial selections are set before execution and remain unmodified during search. This makes the correct calibration of such parameters critical and places emphasis on the user who may not be familiar with the either the search space under investigation or the COGA tool being used.

This chapter addresses these problems by presenting a method where HiCOGA and SsCOGA are used in a closed loop configuration (figure 6.1). A convergence measure is extracted during search and “fed back” into the evolving algorithm, the degree of Halton injection or spatial selection is then adjusted in relation to the convergence measure. The practice of changing such parameters during search is referred to as adaption, and is discussed in the following section.

![Figure 6.1: Open loop COGA](image1.png)

![Figure 6.2: Closed loop COGA](image2.png)
6.2 Adaptive Genetic Algorithms

Adapation within Genetic Algorithms is split into two classes, static and dynamic [Hinterding et. al. 1997 & Eiben et. al. 1999]. Static adaption occurs when the search parameters (population size, probability of selection, crossover, mutation etc.), are set before search and do not change throughout the duration of search. Dynamic adaptation uses mechanisms to alter the search parameters during the run of the GA; and is further split into deterministic, self-adaptive and adaptive groups.

As its name suggests, deterministic dynamic adaption uses deterministic rules to modify the search parameters, and does not receive any feedback information from the GA. Fogarty [Fogarty 1989], gives an early example of deterministic adaption, where the mutation probability decreases during search in relation to a predefined exponential function. In the context of this thesis, more familiar examples are the variable mutation scheme of vmCOGA, the variable incest prevention of HUX / IP and the SsCOGA and HiCOGA algorithms of the previous chapter.

A more sophisticated approach is self-adaptive dynamic adaption, where the search parameters are encoded directly onto the chromosome and therefore evolve in conjunction with the design parameters. The optimal parameter settings are therefore “discovered” during search. Smith [Smith et. al. 1996] encodes the mutation rate into the chromosome of a steady stage GA. Further research has investigated self-adapting the precision of encoding during search [Kim et. al. 1997]. This work develops the hypothesis suggested by Schraudolph [Schraudolph et. al. 1992] i.e. that in the early stages of search a crude encoding precision should be used but as search progresses and the population converges, the precision may be increased to facilitate more concentrated local search. Consequently, the bit length of the chromosome is encoded into the genotype and may increase or decrease during search, and modify the corresponding resolution of the search space.
The third approach and the method used to adapt both Halton injection and spatial selection algorithms is cumbersomely referred to as adaptive dynamic adaption, and will be henceforth referred to as dynamic adaption. This involves the use of feedback information from the evolving process to determine the degree of search parameter modification. Davis [Davis 1989] updates the probability of using specific crossover and mutation mechanisms depending upon the increase or decrease in GA performance. In this manner the more successful operators are more likely to be used during search. A further approach uses a quality criterion to continuously modify the population size of competing subpopulations [Schlierkamp-Voosen et. al. 1996]. The probability of mutation itself is controlled by Lis [Lis 1996]. This is achieved by using a dispersion metric to quantify the proportion of the population that is located about the optimum fitness. The metric is then used to increase or decrease the probability of mutation in incremental steps. It should also be noted that dynamic adaption is not limited to static search environments and has been applied to non-stationary test functions. In one example a mechanism called triggered hypermutation increases the probability of mutation if the “quality” of solutions within the population drops below a certain threshold [Cobb 1990].

From this overview of adaption, and more specifically dynamic adaption, the utility of such methods is clearly apparent. Not only is the need to set search parameters eliminated but the automated re-tuning of such parameters throughout search promotes both the attainment and maintenance of optimal performance. However, a drawback of dynamic adaption is the need to define feedback measures and the mechanism to alter the search parameters. The feedback measure used to control HiCOGA and SsCOGA is based upon the degree of convergence within the population and is discussed in the following section.
6.3 The Feedback Measure – Population Convergence

There are two feedback measures that may be used when adaptively controlling GA search, population convergence and the "success" of search. The so called "1/5 rule" measures the success of an operator [Rechenberg 1973]. It is suggested that the optima rate of convergence $\varphi$, occurs when the ratio of successful\(^1\) mutations to all mutations is 1/5, the mutation rate being modified during search to achieve this. The performance measure suggested by Davis [Davis 1989] relates to the proportion by which a new individual exceeds the fitness of the best individual within the population. An absolute measure is used to control the incest threshold in the CHC algorithm [Eshelman et. al. 1991], in this example the threshold is reduced when there are no children that survive into the next generation. The quality criteria [Schlierkamp-Voosen et. al. 1996] increases the size of a subpopulation (at the expense of the remaining subpopulations) if it consistently produces the fittest individual over the last $w$ generations.

The first approach to explicitly measure population convergence was proposed by DeJong [DeJong 1975]. Here an allele is said to have converged if 95% of the population shares the same individual bit value. Baker [Baker 1985] develops this technique by calculating the percentage of the alleles with the same bit value (50% represents a uniform, unconverted allele, whilst 100% represents a completely converged allele). However, it is clear that in the context of COGA, these techniques are inappropriate since multiple high performance regions may not relate to the same bit value at any particular bit position. The dispersion metric introduced in section 6.2, overcomes these problems by using a measure of population fitness [Lis 1996]. Firstly the fitness distribution of the population is ranked in descending order, if $f_i$ is the fitness of the $i$th population member, the dispersion metric $\rho$ is given by,

\[^1\] A successful mutation is one that increases the fitness of a chromosome [Bäck 1992].
Population dispersion is basically a measure of the standard deviation of the fittest 50% of the population, which is normalised relative to the optimal population fitness. A converged population has a dispersion metric of zero, whilst the maximum dispersion metric is 0.5.

When defining a feedback metric for algorithms like HiCOGA and SsCOGA (specifically designed to increase the sampling of the search space) it is prudent to use a measure based upon the spatial diversity of the population. Such distance metrics have been widely used in many crowding algorithms and also the CHC algorithm to determine the closeness between two chromosomes. Literature also cites their application to both combinatorial problems [Mattfeld 1995] and Genetic Programming [O’Reilly 1997]. The metric presented here develops the dispersion metric, used in both section 5.2.2 to assess the sampling of the transformed leaped Halton sequence and section 5.3.2 to assign a spatial fitness to SsCOGA. To briefly recap, as search converges the overall separation of the population diminishes [Shine et. al. 1997]. The degree of convergence may therefore be measured by calculating the average nearest neighbour Euclidean distance of the population ($E_{nn}$); the decoded population being firstly mapped to a unit hypercube to maintain consistency across all dimensions. $E_{nn}$ is then standardised to the range $\{0, 1\}$ relative to the maximum average nearest neighbour Euclidean distance found during search ($E_{nn}^{max}$). Although not proven, it is believed that $E_{nn}^{max}$ occurs at initialisation. The convergence of the population $C_{nn}$ is then given by,

$$C_{nn} = 1 - \frac{E_{nn}}{E_{nn}^{max}} = \frac{E_{nn}^{max} - E_{nn}}{E_{nn}^{max}}$$
Within a totally unconverged population $\overline{E}_m = E_{\text{max}}^m$ and $C_m = 0.0$, as the population converges $\overline{E}_m \to 0$ and $C_m \to 1.0$. Larger values of $C_m$ therefore indicate higher levels of population convergence.

Once $C_m$ has been determined, the convergence rate ($\dot{C}_m$) at any instance may be found by calculating the derivative of $C_m$ with respect to the number of generations,

$$\dot{C}_m = \frac{dC_m}{d(\text{generation})} \quad ... 6-3$$

$\dot{C}_m$ corresponds to the gradient of the convergence measure profile ($C_m$ vs. generation – section 6.3.1).

Figures 6.3 - 6.6 give the convergence measure profile for different search scenarios when applied to four test functions of increasing dimensionality, TF2, the 6d test function, TBCOM and Keane's 20d bump function (Appendix E). The mutation vectors used to induce each search scenario are given in table 6.1.

<table>
<thead>
<tr>
<th>Search profiles</th>
<th>Mutation vector, $m$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal search</td>
<td>{0.01, 0.01, 0.01, 0.01, 0.01}</td>
</tr>
<tr>
<td>vmCOGA search</td>
<td>{0.08, 0.06, 0.04, 0.02, 0.01}</td>
</tr>
<tr>
<td>Premature convergence</td>
<td>{0.00, 0.00, 0.00, 0.00, 0.00}</td>
</tr>
<tr>
<td>Random walk</td>
<td>{0.20, 0.20, 0.20, 0.20, 0.20}</td>
</tr>
</tbody>
</table>

Table 6.1: The search profiles

Upon examination of figures 6.3 to 6.6 it can be seen that "normal search" is characterised by rapid initial convergence followed by a sustained period of much slower convergence. Premature convergence produces a similar initial period of rapid convergence (albeit of greater magnitude) with the reduction in $C_m$ occurring only with the onset of stagnation (this being more evident in the higher dimensionality functions). As expected, very little convergence is noted during random walk. The most interesting results are obtained
during vmCOGA search, here the majority of convergence occurs when the mutation probability drops at the end of each search stage.

A further characteristic of all four plots is a general decrease in $C_m$ as dimensionality increases. The reason for this is not immediately apparent, however upon inspection of the convergence measure equation, a decrease in $C_m$ must be caused by an increase in the quotient $\bar{E}_{na} / E_{na}^{\text{max}}$. This intern is caused by either a gradual decrease in $E_{na}^{\text{max}}$ or an increase in $\bar{E}_{na}$. As the former explanation is clearly impossible, a disproportionate increase in $\bar{E}_{na}$ with increasing dimensionality forms the only feasible explanation.
Nevertheless, even with the general reduction in $C_m$, the relative nature of the plots remains surprisingly constant as dimensionality increases.

### 6.3.1 Calculating the Convergence Rate

One of the more straightforward ways of calculating the convergence rate $\dot{C}_m$, is to determine the change in convergence measure over successive generations and then divide by two i.e.,

$$\dot{C}_m = \frac{dC_m}{d\text{ (generation)}} = \frac{C_{m_i} - C_{m_{i-1}}}{2} \quad \ldots 6-4$$

Genetic search by its stochastic nature is noisy and the above method may increase the sensitivity of control due to fluctuations in $C_m$ during generations. This problem has been reported by Lis when calculating the dispersion rate [Lis 1996]. Calculating the overall trend of the convergence measure reduces the effects of such fluctuations and is achieved by finding the line of best fit (regression line) over the previous $n$ generations. Figure 6.7 gives an example of calculating $\dot{C}_m$ at generation 17 by regressing the values of $C_m$ over the preceding four generations (including generation 17).

<table>
<thead>
<tr>
<th>$i$ (n=4)</th>
<th>gen$_i$</th>
<th>$C_m$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>14</td>
<td>$C_m$ at generation 14</td>
</tr>
<tr>
<td>2</td>
<td>15</td>
<td>$C_m$ at generation 15</td>
</tr>
<tr>
<td>3</td>
<td>16</td>
<td>$C_m$ at generation 16</td>
</tr>
<tr>
<td>4</td>
<td>17</td>
<td>$C_m$ at generation 17</td>
</tr>
</tbody>
</table>

Figure 6.7: Calculating $\dot{C}_m$

The drawback of this technique is the need to set a value for $n$. To increase accuracy, $n$ should be as small as possible, however if $n$ is too small over sensitivity to spurious fluctuations in $C_m$ may occur. The effect of increasing $n$ is given in figure 6.8, this shows the convergence rate profile for vmCOGA when applied to the 6d test function (the corresponding convergence measure response is given in figure 6.4). The dotted black
stepped line is obtained by independently regressing across the 25 generations of each search stage and gives a more accurate prediction of the true convergence rate.

![Graph showing regression points and true convergence rate]

**Figure 6.8:** Setting the number of regression points

At smaller values of $n$, the response at the start of each search stage is quicker, however the "noisy" appearance of the profile is greater. As $n$ increases the curve noise decreases but the response becomes noticeably more sluggish. Of the four investigated values of $n$, a value of 6 appears to offer the best compromise between robustness and response.

### 6.4 The Control Mechanism

The most common approach used to control a process is referred to as proportional control [Parr 1996], here the change in control input is proportional to the error, $\Delta$ between the target output, $O_t$ and the actual output, $O_a$,

$$\Delta = O_t - O_a$$

...6-5

Hence, $\Delta$ is the difference between the target convergence rate, $\dot{C}_{t/m}$ and the actual convergence rate, $\dot{C}_{m}$,

$$\Delta = \dot{C}_{t/m} - \dot{C}_{m}$$

...6-6
The control input ($I_i$) is increased or decreased by adding $k\Delta$ to the previous control input ($I_{i-1}$), ($\Delta$ is a vector quantity and $k$ is referred to as the proportional gain),

$$I_i = I_{i-1} + k\Delta$$

In the context of HiCOGA and SsCOGA search there are two drawbacks when using proportional control. Firstly the number of Halton injections and spatial selection are discrete parameters (since it is impossible to make half a spatial selection). To overcome this, the parameters are redefined as Halton injection ($r_{Hi}$) and spatial selection ($r_{ss}$) rates,

$$n_{Hi} = \left\lfloor r_{Hi} \times n_{cross} \right\rfloor$$
$$n_{ss} = \left\lfloor r_{ss} \times n_{cross} \right\rfloor$$

where,

$n_{Hi}$ = number of Halton injections per generation [$0, n_{cross}$]

$n_{ss}$ = number of spatial selections per generation [$0, n_{cross}$]

$r_{Hi}$ = the Halton injection rate [0, 1]

$r_{ss}$ = the spatial selection rate [0, 1]

$n_{cross}$ = the number of chromosomes selected for reproduction

[] = rounding process to the nearest integer value

$r_{Hi}$ and $r_{ss}$ therefore define the proportion of reproduced individuals per generation that are Halton injections or spatial selections.

The second drawback is the need to set the proportional gain. This is more problematic since it must occur before control may commence. Traditionally, $k$ is calibrated by using methods such as Ziegler-Nichols [Ziegler et. al. 1942]. This technique tunes $k$ during a test run of the process, and as such has proven to be unsuitable for use with either HiCOGA or SsCOGA.
Instead, \( k \) is determined\(^2\) by considering both the maximum expected error (\( \Delta_{\text{max}} \)) and the maximum allowable change in \( r_{\text{Hi}} \) or \( r_{\text{ss}} \) (\( \delta r_{\text{Hi,ss}} \)). With reference to figure 6.9, it can be seen that the maximum increase in \( \delta r_{\text{Hi,ss}} \) corresponds to the maximum negative error, similarly the largest decrease increase in \( \delta r_{\text{Hi,ss}} \) coincides with largest negative error. It follows that the proportional gain is represented by the gradient of the curve in the region 

\[-\Delta_{\text{max}} < \Delta < +\Delta_{\text{max}},\]

\[k = -\frac{\delta r_{\text{Hi,ss}}}{\Delta_{\text{max}}}.\] \( ...6.9 \)

The regions \(-\Delta_{\text{max}} \geq \Delta \geq +\Delta_{\text{max}}\) relates to control actions that produce a constant change in \( r_{\text{Hi}} \) or \( r_{\text{ss}} \) that is independent of the actual error and therefore reduces the possibility of over-sensitive control for large \( \Delta \).

Figure 6.9: Determining the control gain

\(^2\) It should be noted that the value for \( k \) has not been designed to promote optimal control response either transient or steady state

\(^3\) \( \delta r_{\text{Hi,ss}} \) denotes \( \delta r_{\text{Hi}} \) and \( l \) or \( \delta r_{\text{ss}} \).
The proportional action used to control HiCOGA and SsCOGA at the $i$th generation is represented by,

$$r_{Hi} = r_{Hi-1} + \delta r_{Hi}$$
$$r_{ss} = r_{ss-1} + \delta r_{ss}$$

Where $\delta r_{Hi,ss}$ for any $\Delta$ may be found from figure 6.9. To maintain consistency, $\delta r_{Hi,ss}$ and $\delta r_{ss}$ are taken as 10% of their respective operating ranges,

$$\delta r_{Hi,max} = 0.1 r_{Hi,max}$$
$$\delta r_{ss,max} = 0.1 r_{ss,max}$$

### 6.4.1 Defining the Maximum Target Rate Error

When one is searching for the maximum error, the ideal approach would be to investigate every possible search profile and target convergence rate combination, however this is clearly infeasible. Nevertheless, it is clear that the maximum error arises when the target rate ($\dot{C}_m$) is at its smallest and the true convergence rate ($\dot{C}_m$) is at its maximum. This occurs during premature convergence and when $\dot{C}_m = 0$ (the $C_m$ vs. generation plots of figures 6.3 - 6.6 and the $\dot{C}_m$ vs. generation plot of figure 6.8 all substantiate this hypothesis).

Figure 6.10 gives the average error taken over 100 independent trials upon the 6d-test function where premature convergence is induced by using a zero mutation vector, $m = \{0.00, 0.00, 0.00, 0.00, 0.00\}$. It can be seen that the maximum instantaneous error ($\Delta_{max} = 0.01846$) occurs at the earliest stages of search. At later stages of search the error reduces as search stagnates and $\dot{C}_m \rightarrow 0$. However, the maximum error taken from this search scenario is somewhat conservative and may desensitise search, as a result $\Delta_{max}$ is set as 50% of its observed value, 0.00924.
6.4.2 The Adaptive Algorithms

Four adaptive algorithms are investigated. Adaptive mutation COGA (amCOGA) uses mutation alone as the controlled parameter, and is therefore the closest relative to vmCOGA within such an adaptive environment. The second and third test algorithms are adaptive variants of HiCOGA and SsCOGA. The fourth example, mssCOGA, eliminates the need to set the baseline mutation rate within SsCOGA since mutation is adapted along with $r_s$ during search. Table 6.2 lists the test algorithms and their corresponding control parameter values. All control parameters are initialised at their mid-values.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Controlled Parameter</th>
<th>range</th>
<th>initial value</th>
<th>maximum increment</th>
</tr>
</thead>
<tbody>
<tr>
<td>amCOGA</td>
<td>Mutation</td>
<td>$0.0 \leq p_{\text{mutation}} \leq 0.1$</td>
<td>-</td>
<td>0.01</td>
</tr>
<tr>
<td>HiCOGA</td>
<td>Halton injection</td>
<td>$0.0 \leq r_H \leq 1.0$</td>
<td>0.5</td>
<td>0.1</td>
</tr>
<tr>
<td>SsCOGA</td>
<td>Spatial selection</td>
<td>$0.0 \leq r_s \leq 1.0$</td>
<td>0.5</td>
<td>0.1</td>
</tr>
<tr>
<td>mssCOGA</td>
<td>Mutation and</td>
<td>$0.0 \leq p_{\text{mutation}} \leq 0.1$</td>
<td>0.05</td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td>Spatial selection</td>
<td>$0.0 \leq r_s \leq 1.0$</td>
<td>0.5</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Table 6.2: The adaptive algorithms
Control commences when there is sufficient extracted \( C_m \) data to allow for \( \dot{C}_m \) regression to take place. Before this point, diversity is maintained by mutation alone (search is therefore analogous to vmCOGA during this phase). At the end of the \( n \)-th-1 generation (\( n \) is the number of generations over which \( C_m \) is regressed to determine \( \dot{C}_m \)) the control parameters are reinitialised in accordance with table 6.3. In the case of HiCOGA and SsCOGA, \( p(mutation) \) is also reset to 0 and 0.001 respectively. After the completion of the \( n \)-th generation, \( \dot{C}_m \) is calculated and the first adaption of the control parameter takes place.

<table>
<thead>
<tr>
<th>Gen</th>
<th>amCOGA</th>
<th>HiCOGA</th>
<th>SsCOGA</th>
<th>mssCOGA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( p(\text{mut}) )</td>
<td>( p(\text{mut}) )</td>
<td>( r_{HH} )</td>
<td>( r_{\text{mut}} )</td>
</tr>
<tr>
<td>1</td>
<td>0.08</td>
<td>0.08</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>2</td>
<td>0.08</td>
<td>0.08</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>3</td>
<td>0.08</td>
<td>0.08</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>4</td>
<td>0.08</td>
<td>0.08</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>5</td>
<td>0.08</td>
<td>0.00</td>
<td>0.50</td>
<td>0.00</td>
</tr>
<tr>
<td>6</td>
<td>0.08</td>
<td>0.00</td>
<td>0.50</td>
<td>0.00</td>
</tr>
<tr>
<td>7</td>
<td>( p(\text{mut})^* )</td>
<td>0.00</td>
<td>( r_{HH}^* )</td>
<td>0.001</td>
</tr>
<tr>
<td>8</td>
<td>( p(\text{mut})^* )</td>
<td>0.00</td>
<td>( r_{HH}^* )</td>
<td>0.001</td>
</tr>
</tbody>
</table>

Table 6.3: The control parameter values in the generations preceding and succeeding the commencement of control. \( p(\text{mut})^* \), \( r_{HH}^* \) and \( r_{\text{mut}}^* \) indicate adapted parameters (\( n = 6 \), section 6.3.1)

5.5 Target Convergence Rate Study

To simplify application, the target rates (\( \dot{C}_m' \)) used during the convergence rate study are based upon the vmCOGA convergence rate (\( \dot{C}_m^{vm} \)). For the 6d-test function this is found by regressing \( C_m \) over all the 125 generations of figure 6.11. \( \dot{C}_m^{vm} \) is found to be 0.00195 generation\(^{-1}\).

The following sections present the results of the convergence rate study, in each case the target rate is a multiple of \( \dot{C}_m^{vm} \) (1, 1.5 and 2), the generation and filtering vectors concur with those of section 4.7. The RD results are presented in terms of regional sensitivity rank as defined in section 5.4.1. The number of regions where vmCOGA is outperformed within each sensitivity group measures the performance relative to vmCOGA (tables 6.5, 119...
6.7 and 6.9). These tables also give RD performance relative to the four test algorithms, this is achieved by ranking the regional decomposition of each algorithm (one indicates the highest value in the region whilst four indicates the lowest value). The RD rankings are then averaged over the six high performance regions within each sensitivity group.

![Figure 6.11: Calculating $\hat{C}_m$ for vmCOGA on the 6d test function](image)

### 6.5.1 Target Rate = $\hat{C}_m^{vm}$

When the target convergence rate is set to the vmCOGA rate found in figure 6.11 (0.00195 generation⁻¹), the GDE and FCSₚₚₑ values for each test algorithm are similar to the control vmCOGA algorithm, suggesting comparable levels of convergence within the FCS (table 6.4). However, in each test case (excluding mssCOGA) a reduction in set cover is noted, this being more prevalent in HiCOGA. The general increase in GDE standard deviation in HiCOGA, SsCOGA and mssCOGA is discussed in section 5.4.2.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Set cover</th>
<th>GDE</th>
<th>FCSₚₚₑ</th>
<th>FCSₚₚₑ</th>
</tr>
</thead>
<tbody>
<tr>
<td>vmCOGA</td>
<td>285.72</td>
<td>42.11% (30.16%)</td>
<td>1.344</td>
<td>1.471</td>
</tr>
<tr>
<td>amCOGA</td>
<td>270.54</td>
<td>44.98% (28.41%)</td>
<td>1.345</td>
<td>1.470</td>
</tr>
<tr>
<td>HiCOGA</td>
<td>208.80</td>
<td>41.81% (31.59%)</td>
<td>1.340</td>
<td>1.468</td>
</tr>
<tr>
<td>SsCOGA</td>
<td>269.93</td>
<td>44.33% (44.06%)</td>
<td>1.340</td>
<td>1.467</td>
</tr>
<tr>
<td>mssCOGA</td>
<td>287.25</td>
<td>42.12% (37.68%)</td>
<td>1.340</td>
<td>1.469</td>
</tr>
</tbody>
</table>

**Table 6.4: FCS results ($\hat{C}_m = \hat{C}_m^{vm}$)**

(values in parenthesis are standard deviations expressed as a % of measured value)
When comparing the regional decomposition results of each algorithm (figure 6.12 and left hand section of table 6.5), large differences in the ability of each test algorithm to outperform the control algorithm within the high and low sensitivity groups is noted. If amCOGA is discussed initially, an increase in performance in the low sensitivity regions is coupled with a decrease in performance within the high sensitivity regions; this further supports the hypothesis that mutation alone is an ineffective means of promoting the exploration of sensitive regions. In support of the findings of chapter 5, SsCOGA performs in the opposite manner, here high sensitivity performance increases whilst low sensitivity performance decreases. MssCOGA with its mix of mutation and spatial selection offers a degree of compromise. The performance of HiCOGA is disappointing and is attributable to a reduction in set cover, nevertheless when improvement over vmCOGA does occur it is in the high and medium sensitivity groups suggesting some increased exploration of such regions. In each test case excluding HiCOGA, vmCOGA is outperformed within 44.44% of all high performance regions.

The relative performance ranking of the test algorithms (right-hand side of table 6.5) generally substantiates the findings of the previous paragraph. Within the high sensitivity
group, SsCOGA receives the highest rank whilst amCOGA possesses the lowest. Conversely, within the low sensitivity group, amCOGA has the highest performance rank and SsCOGA one of the lowest (only HiCOGA has a poorer rank). Again mssCOGA appears to provide a compromise between both amCOGA and SsCOGA.

<table>
<thead>
<tr>
<th></th>
<th>Number of regions where vmCOGA is outperformed</th>
<th>Average performance rank (1-4)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>High</td>
<td>Medium</td>
</tr>
<tr>
<td>amCOGA</td>
<td>2/6</td>
<td>2/6</td>
</tr>
<tr>
<td>HiCOGA</td>
<td>2/6</td>
<td>1/6</td>
</tr>
<tr>
<td>SsCOGA</td>
<td>4/6</td>
<td>3/6</td>
</tr>
<tr>
<td>mssCOGA</td>
<td>2/6</td>
<td>4/6</td>
</tr>
</tbody>
</table>

Table 6.5: Relative RD ($\hat{C}_m' = \hat{C}_{\text{vm}}$)
(Low, medium and high relate to sensitivity groupings)

6.5.2 Target Rate = $1.5 \hat{C}_n'$

As the target convergence rate increases to a value 1.5 times that of vmCOGA (0.002925 generation), increases in GDE and $FCS_{\text{ave}}$ indicate higher levels of convergence within the FCS of all test algorithms (table 6.6). The reduction in GDE standard deviation when compared with the lower target rate is caused by a reduction in the mutation, spatial selections and Halton injections made during search as the target convergence increases. This intern reduces overzealous exploration of low performance regions and its associated effects upon GDE standard deviation (section 5.4.2)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Set cover</th>
<th>GDE</th>
<th>$FCS_{\text{ave}}$</th>
<th>$FCS_{\text{best}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>vmCOGA</td>
<td>285.72 (28.00%)</td>
<td>42.11% (30.16%)</td>
<td>1.344</td>
<td>1.471</td>
</tr>
<tr>
<td>amCOGA</td>
<td>276.65 (31.17%)</td>
<td>58.58% (28.99%)</td>
<td>1.362</td>
<td>1.474</td>
</tr>
<tr>
<td>HiCOGA</td>
<td>212.48 (31.53%)</td>
<td>55.46% (30.99%)</td>
<td>1.359</td>
<td>1.471</td>
</tr>
<tr>
<td>SsCOGA</td>
<td>249.58 (40.08%)</td>
<td>56.87% (36.28%)</td>
<td>1.359</td>
<td>1.468</td>
</tr>
<tr>
<td>mssCOGA</td>
<td>271.26 (30.31%)</td>
<td>56.62% (27.58%)</td>
<td>1.358</td>
<td>1.474</td>
</tr>
</tbody>
</table>

Table 6.6: FCS results ($\hat{C}_m' = 1.5\hat{C}_n'$)
(values in parenthesis are standard deviations expressed as a % of measured value)

The inability of amCOGA to support sustained exploration of the most sensitive regions (noted in section 6.5.1) becomes more evident in figure 6.13 and table 6.7. In all regions of high sensitivity, amCOGA is outperformed by vmCOGA, conversely in the more robust
regions amCOGA outperforms vmCOGA as search converges upon the lower sensitivity regions within the search space. Such convergence is not apparent in the results of the remaining test algorithms, where HiCOGA, SsCOGA and mssCOGA outperform vmCOGA in five, six and four of the six high performance regions respectively. These results suggest that Halton injection and spatial selection may be used during search in conjunction with higher target convergence rates without the loss of sensitive region exploration. Furthermore, in the case of mssCOGA and more notably SsCOGA a significant increase in performance within the less sensitive regions results in both algorithms outperforming vmCOGA in 83.33% and 94.4% of all HP regions respectively. The results of HiCOGA are again disappointing, since vmCOGA is outperformed in only one of the low sensitivity regions, this may be due to the overly disruptive nature of Halton injection suggested in section 5.4.2, coupled with a significant decrease in FCS size.

The relative nature of the performance rankings discussed in section 6.5.1 are still evident at this higher target convergence rate, however the spread of the average ranking reduces (final column table 6.7). Unsurprisingly, amCOGA is the lowest ranked algorithm in the most sensitive regions but the highest in the least sensitive regions. Again SsCOGA
receives the highest rank in the sensitive regions and one of the lowest in the least sensitive regions. Nevertheless, SsCOGA offers the best combined performance across all regions of differing sensitivity.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Number of regions where vmCOGA is outperformed</th>
<th>Average performance rank (1-4) when compared with 4 test algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>High</td>
<td>Medium</td>
</tr>
<tr>
<td>amCOGA</td>
<td>0/6</td>
<td>5/6</td>
</tr>
<tr>
<td>HiCOGA</td>
<td>5/6</td>
<td>3/6</td>
</tr>
<tr>
<td>SsCOGA</td>
<td>6/6</td>
<td>5/6</td>
</tr>
<tr>
<td>msCOGA</td>
<td>4/6</td>
<td>6/6</td>
</tr>
</tbody>
</table>

Table 6.7: Relative RD ($\hat{C}_m = 1.5 \hat{C}_m^{\text{vm}}$)

(Low, medium and high relate to sensitivity groupings)

6.5.3 Target Rate =$2\hat{C}_m^{\text{vm}}$

In the final test example, where the target convergence rate is twice that of vmCOGA (0.0039 generations$^{-1}$), there is a noticeable drop in RD performance within the most sensitive regions. It follows that at such extreme rates of convergence the overall degree of Halton injection and spatial selection diminishes further. As a result, the ability of HiCOGA, SsCOGA and msCOGA to explore the more sensitive regions starts to reduce and gives way to convergence upon the less sensitive regions (in every region within the low sensitivity group and the majority of the medium sensitivity groups, vmCOGA is outperformed). However, even at such excessive convergence rates, SsCOGA outperforms vmCOGA within half of the most sensitive regions (figure 6.14 and table 6.9).

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Set cover</th>
<th>GDE</th>
<th>FCS$_{avg}$</th>
<th>FCS$_{best}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>vmCOGA</td>
<td>285.72 (28.00%)</td>
<td>42.11% (30.16%)</td>
<td>1.344</td>
<td>1.471</td>
</tr>
<tr>
<td>amCOGA</td>
<td>277.57 (30.10%)</td>
<td>65.66% (22.19%)</td>
<td>1.375</td>
<td>1.477</td>
</tr>
<tr>
<td>HiCOGA</td>
<td>235.56 (37.52%)</td>
<td>64.02% (25.80%)</td>
<td>1.371</td>
<td>1.475</td>
</tr>
<tr>
<td>SsCOGA</td>
<td>266.32 (38.00%)</td>
<td>63.50% (31.63%)</td>
<td>1.368</td>
<td>1.472</td>
</tr>
<tr>
<td>msCOGA</td>
<td>273.50 (31.34%)</td>
<td>63.43% (26.96%)</td>
<td>1.372</td>
<td>1.475</td>
</tr>
</tbody>
</table>

Table 6.8: FCS results ($\hat{C}_m = 2\hat{C}_m^{\text{vm}}$)

(values in parenthesis are standard deviations expressed as a % of measured value)

As in the two previous examples, the relative performance ranking of each algorithm is still maintained, however the reduction in the spread of the average ranking, noted in the previous section, diminishes further.

124
The findings of the convergence study upon the 6d-test function appears to support the conclusions of the previous chapter, namely that the actions of spatial selection and Halton injection are able to increase the exploration of the more sensitive regions within the search space. Furthermore, with convergence rates equal to that of vmCOGA, increased exploration of one sensitivity group appears to be at the detriment of exploration within the opposing sensitivity group (the results of amCOGA and SsCOGA in section 6.5.1 support this suggestion in dichotomous ways). In this instance, the use of adaptive mutation and spatial selection within the mssCOGA algorithm appears to provide a compromise between amCOGA and SsCOGA. As the convergence rate increases, amCOGA performance within the most sensitive regions drops. This phenomenon is not immediately evident in the remaining test algorithms, where high levels of RD performance are noted across the
majority of high performance regions. However if target rate is increased further, performance within the most sensitive regions drops as search converges upon the low and medium sensitivity regions. In this investigation it appears that a convergence rate of 1.5 times that of vmCOGA provides optimal performance.

### 6.6 Application to Additional Test Functions: The Calibration Curves

The target convergence rates investigated in the previous sections are based upon that of vmCOGA search when applied to the 6d-test function for 25 generations / search stage. Although the appearance of the vmCOGA convergence profile changes little with increasing dimensionality (figures 6.3-6.6), there is a gradual decrease in the convergence rate caused by an increase in the number of generations during search. It follows that the vmCOGA rate calculated in section 6.5 is unsuitable for use with objective functions of higher and lower dimensionality. Furthermore, it is unwise to assume that for any given dimensionality, $\hat{C}_m$ remains constant as the number of generations / search stage increases.

![Figure 6.15: vmCOGA calibration curves](image-url)
To overcome this, the convergence rates for vmCOGA search when applied to TF1, TF2, the 6d-test function, miniCAPS, TBCOM and Keane’s 20d-bump function are found for 11 search cases of increasing duration (5, 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100 generations / search stage). The approach outlined in section 6.5 is used to calculate the resulting convergence rate in each case (this rate being averaged over 100 independent trials). Each line within figure 6.15 represents the vmCOGA convergence rates for each of the six test functions. These curves are used to estimate the vmCOGA convergence rate for any test function (dimensionality between 2 and 20) and for any search duration (between 5 and 100 generations / search stage).

Polynomial regression analysis may be applied to give the equation of each line and hence the relationship between $\hat{C}_m$ and generations / search stage. For example, the vmCOGA convergence rate for the 6d test function was found to be (8\textsuperscript{th} order polynomial regression),

$$\hat{C}_m = 1.41 \times 10^{-16} \text{gen}^8 - 6.88 \times 10^{-14} \text{gen}^7 + 1.43 \times 10^{-11} \text{gen}^6 - 1.65 \times 10^{-9} \text{gen}^5 + 1.16 \times 10^{-7} \text{gen}^4 - 5.03 \times 10^{-6} \text{gen}^3 + 2.0 \times 10^{-3} \text{gen}^2 - 2.0gen + 0.0152 \quad \text{...6-12}$$

where gen = generations / search stage

### 6.6.1 Application to TF1 and TF2

This section investigates the application of the four adaptive algorithms to the lower dimensionality test functions, TF1 and TF2, of section 4.4. In both cases, the generation and filtering vectors were the same as those first used in section 4.2.1,

$$g = \{5, 10, 15, 20, 25\}$$

$$Rf = \{0.75, 0.75, 0.50, 0.25, 0.00\}$$

The control mechanism and its settings (maximum error, number of interpolation points, initial control values etc.) are the same as those used for the 6d-convergence study. The
required vmCOGA convergence rate is taken from the calibration curves of figure 4.15. The slight difference between the vmCOGA rates relating to TF1 and TF2 are averaged into a single value that is applied to both test functions ($C_{vm} = 0.00688$ generations$^{-1}$). The final clustering set and regional decomposition results for TF1 and TF2 are given in tables 6.11-6.12 and figures 6.16-6.19 respectively. The regions of high performance are again presented in terms of descending regional sensitivity ranking (table 6.10).

<table>
<thead>
<tr>
<th>Regional</th>
<th>Region</th>
<th>TF1 Number of solutions in region (vmCOGA)</th>
<th>Region</th>
<th>TF2 Number of solutions in region (vmCOGA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>sensitivity rank</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>39.16</td>
<td>1</td>
<td>45.13</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>42.81</td>
<td>2</td>
<td>57.24</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>46.30</td>
<td>5</td>
<td>85.58</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>47.33</td>
<td>4</td>
<td>97.04</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>50.75</td>
<td>3</td>
<td>207.82</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>92.11</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>101.37</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 6.10: Regional sensitivity ranking TF1 and TF2

<table>
<thead>
<tr>
<th>$\hat{C}_{m}^s$</th>
<th>Algorithm</th>
<th>Set cover</th>
<th>GDE</th>
<th>FCS$_{avg}$</th>
<th>FCS$_{best}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>vmCOGA 586.46 (7.84%)</td>
<td>71.78% (10.35%)</td>
<td>1.081</td>
<td>1.227</td>
<td></td>
</tr>
<tr>
<td></td>
<td>amCOGA 592.36 (11.05%)</td>
<td>70.88% (8.59%)</td>
<td>1.079</td>
<td>1.228</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HiCOGA 468.91 (8.26%)</td>
<td>71.77% (10.36%)</td>
<td>1.081</td>
<td>1.226</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SsCOGA 475.71 (8.95%)</td>
<td>70.59% (11.77%)</td>
<td>1.079</td>
<td>1.226</td>
<td></td>
</tr>
<tr>
<td></td>
<td>mssCOGA 534.20 (11.23%)</td>
<td>70.91% (10.47%)</td>
<td>1.079</td>
<td>1.227</td>
<td></td>
</tr>
<tr>
<td>$1.5\hat{C}_{m}^v$</td>
<td>amCOGA 523.30 (10.31%)</td>
<td>75.26% (11.33%)</td>
<td>1.086</td>
<td>1.228</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HiCOGA 463.02 (7.88%)</td>
<td>75.50% (9.54%)</td>
<td>1.085</td>
<td>1.227</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SsCOGA 468.91 (9.13%)</td>
<td>75.64% (12.24%)</td>
<td>1.086</td>
<td>1.227</td>
<td></td>
</tr>
<tr>
<td></td>
<td>mssCOGA 496.75 (11.21%)</td>
<td>74.73% (12.52%)</td>
<td>1.085</td>
<td>1.228</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.11: FCS results for TF1 study
(values in parenthesis are standard deviations expressed as a % of measured value)

<table>
<thead>
<tr>
<th>$\hat{C}_{m}^s$</th>
<th>Algorithm</th>
<th>Set cover</th>
<th>GDE</th>
<th>FCS$_{avg}$</th>
<th>FCS$_{best}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>vmCOGA 560.35 (5.49%)</td>
<td>87.93% (3.95%)</td>
<td>0.741</td>
<td>0.915</td>
<td></td>
</tr>
<tr>
<td></td>
<td>amCOGA 565.17 (7.36%)</td>
<td>86.77% (4.19%)</td>
<td>0.738</td>
<td>0.916</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HiCOGA 438.04 (6.00%)</td>
<td>86.96% (5.45%)</td>
<td>0.740</td>
<td>0.915</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SsCOGA 442.36 (6.54%)</td>
<td>86.68% (5.51%)</td>
<td>0.743</td>
<td>0.914</td>
<td></td>
</tr>
<tr>
<td></td>
<td>mssCOGA 512.24 (7.75%)</td>
<td>86.57% (5.12%)</td>
<td>0.739</td>
<td>0.916</td>
<td></td>
</tr>
<tr>
<td>$1.5\hat{C}_{m}^v$</td>
<td>amCOGA 528.84 (7.66%)</td>
<td>89.05% (4.67%)</td>
<td>0.748</td>
<td>0.915</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HiCOGA 443.11 (6.21%)</td>
<td>88.10% (4.73%)</td>
<td>0.747</td>
<td>0.916</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SsCOGA 446.41 (6.00%)</td>
<td>87.76% (5.73%)</td>
<td>0.745</td>
<td>0.914</td>
<td></td>
</tr>
<tr>
<td></td>
<td>mssCOGA 488.04 (6.51%)</td>
<td>88.342% (4.96%)</td>
<td>0.746</td>
<td>0.914</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.12: FCS characteristics for TF2 study
(values in parenthesis are standard deviations expressed as a % of measured value)
Overall, the regional decomposition results are disappointing. In all high performance regions, HiCOGA, SsCOGA and mssCOGA are outperformed by vmCOGA. Some improvement is offered by amCOGA, outperforming vmCOGA in four of the six and two of the five regions of TF1 and TF2 respectively, however this improvement is present in only the lesser of the two convergence rates and is lost as the rate increases. An explanation for this is a significant reduction in amCOGA set cover when moving from the smaller to the larger target convergence rate. This characteristic is also noted with mssCOGA albeit with much reduced severity. In the case of HiCOGA and SsCOGA, the drop in set cover is marginal (moreover, it increases when applied to TF2) and unlike amCOGA, results in no significant drop in RD as \( \dot{C}_m \) increases.

An explanation for the decrease in overall RD performance when moving from the 6d to the 2d-test function is given by the results of table 6.13. This gives the Relative Set Cover (RSC) for convergence rates of \( \dot{C}_m \) and 1.5 \( \dot{C}_m \) for TF1, TF2 and the 6d-test function. RSC is calculated by dividing the actual set cover by the corresponding set cover for
vmCOGA. Hence, an RSC of 1 indicates comparable FCS sizes between the test and vmCOGA algorithms.

Figure 6.18: Regional decomposition results for TF2 ($\hat{C}_m = C_m^{\text{mn}}$)

Figure 6.19: Regional decomposition results for TF2 ($C'_m = 1.5 C_m^{\text{mn}}$)

Table 6.13: Relative Set cover TF1 and TF2

For both SsCOGA and mssCOGA, RSC is higher when applied to the 6d-test function, but drops notably upon application to TF1 and TF2. It follows that this drop severely reduces RD within the lower dimensionality test function. The HiCOGA results conflict with this trend since RSC increases when applied to TF1 and TF2, however these values are still smaller than SsCOGA and considerably smaller than both amCOGA and mssCOGA. RSC for amCOGA appears to be dependent upon the applied target convergence rate. At a rate equal to vmCOGA, the 2d test functions give the higher RSC values, whilst at the higher convergence rates, the 6d-test function gives the greater RSC.
Although relative set cover provides an explanation for the decrease in SsCOGA and HiCOGA performance when applied to the lower dimensionality functions, it does not discount the ability of either algorithm to increase the exploration of sensitive regions. The following section investigates this by using a regional decomposition measure that is independent of set cover. This is achieved by expressing the RD of figures 6.16-6.19 as a percentage of the total number of solutions in all high performance regions. For example the proportion of high performance solutions lying in region 2 ($P_2$) is given by,

$$ P_2 = \frac{RH_2}{\sum_{i=1}^{N} RH_i} $$

where,

$$ N = \text{number of high performance regions} $$

$$ RH_i = \text{number of solutions in } i\text{th region} $$

As with the regional decomposition results, the proportion results are expressed relative to vmCOGA,

$$ RP_i = \left( \frac{P_i - P_{vmCOGA}}{P_{vmCOGA}} \right) \times 100 $$

where,

$$ RP_i = \text{Relative proportion of the } i\text{th region} $$

$$ P_{vmCOGA} = \text{Relative proportion of the } i\text{th region given by vmCOGA} $$

The results given by the lower target convergence rate ($\dot{C}_m^{\ast} = \dot{C}_m^{vm}$), indicate that within the most sensitive regions (rank 1, 2, 3 and 4 of TF1 and rank 1 and 2 of TF2) there is an increase in RP in all test algorithms (figures 6.20 and 6.22). This provides some evidence of the increased sampling of sensitive regions within the search space. As noted in the 6d study using the smaller target rate, a reduction in RP within the regions of lower sensitivity (rank 5, 6 and 7 of TF1 and rank 3, 4 and 5 of TF2) is also evident. As the target...
convergence rate rises to $1.5 \hat{C}_m^{\text{im}}$, the increased RP within the more sensitive regions is still present within the HiCOGA and mssCOGA results of TF1 (figure 6.21), but absent in amCOGA and greatly reduced with SsCOGA. Moreover, when the increased target rate is applied to TF2 there is little or no evidence to indicate any increase in the exploration of the more sensitive regions in any of the test algorithms.

Figure 6.20: Relative proportion of all HP solutions for TF1 ($\hat{C}_m^{t} = \hat{C}_m^{\text{im}}$)

Figure 6.21: Relative proportion of all HP solutions for TF1 ($\hat{C}_m^{t} = 1.5 \hat{C}_m^{\text{im}}$)

Figure 6.22: Relative proportion of all HP solutions for TF2 ($\hat{C}_m^{t} = \hat{C}_m^{\text{im}}$)

Figure 6.23: Relative proportion of all HP solutions for TF2 ($\hat{C}_m^{t} = 1.5 \hat{C}_m^{\text{im}}$)
Treated in isolation, the RD results from the 2d study appear to contradict the findings of the 6d study, however upon closer inspection this is not the case. By using the relative proportion measure, it has been shown that the ability of Halton injection and Spatial selection to increase the sampling of sensitive regions is still present (albeit at a lower target convergence rate than that in the 6d study). Nevertheless, characteristically smaller set cover severely limits the absolute regional decomposition abilities of HiCOGA, SsCOGA and mssCOGA.

It is therefore suggested that in the extremely unlikely event that in a real world engineering environment, 2d test functions are to be investigated in any practical context, vmCOGA, amCOGA (with a target convergence rate equal to vmCOGA) or the 30x30 ECOGA of chapter 4 should be used.

6.6.2 Application miniCAPS, TBCOM and Keane’s 20d Bump Function

The lack of knowledge relating to the location and nature of the HP regions within the search domains described by miniCAPS, TBCOM and Keane’s 20d test functions makes it impossible to thoroughly investigate the performance under conditions of higher dimensionality (a problem that is discussed in section 4.6). Although values for GDE and RD may not be determined, FCSave and set cover can be calculated and may be used to compare both convergence within the FCS and RSC and to indicate if the reduced RSC, which caused poor RD performance within the 2d functions, is present upon application to higher dimensionality functions.

<table>
<thead>
<tr>
<th>Test function</th>
<th>FCSave</th>
<th>Set cover</th>
<th>MiniCAPS</th>
<th>TBCOM</th>
<th>Keane</th>
<th>miniCAPS</th>
<th>TBCOM</th>
<th>Keane</th>
</tr>
</thead>
<tbody>
<tr>
<td>vmCOGA</td>
<td>9327.26</td>
<td>0.000941</td>
<td>0.217</td>
<td>202.77</td>
<td>583.56</td>
<td>7257.21</td>
<td></td>
<td></td>
</tr>
<tr>
<td>amCOGA</td>
<td>9350.60</td>
<td>0.000949</td>
<td>0.217</td>
<td>194.51</td>
<td>579.62</td>
<td>6607.59</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HiCOGA</td>
<td>9244.21</td>
<td>0.000935</td>
<td>0.247</td>
<td>122.47</td>
<td>414.99</td>
<td>5141.23</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SsCOGA</td>
<td>9241.69</td>
<td>0.000923</td>
<td>0.260</td>
<td>324.45</td>
<td>643.52</td>
<td>6873.88</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mssCOGA</td>
<td>9279.65</td>
<td>0.000967</td>
<td>0.196</td>
<td>241.37</td>
<td>552.39</td>
<td>6575.27</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 6.14: Average FCS fitness (miniCAPS is set to maximise ferry range)(\(\hat{C}_m = C_{nm}\))
Nevertheless, it should be noted that the absence of GDE and RD data limits the degree of confidence one may place upon the findings of such a study, since the increased exploration of sensitive regions cannot be verified. Consequently, these preliminary results should be treated with caution and corroborated by a more exacting study.

<table>
<thead>
<tr>
<th></th>
<th>Test function</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TF1</td>
</tr>
<tr>
<td>amCOGA</td>
<td>1.010</td>
</tr>
<tr>
<td>HiCOGA</td>
<td>0.800</td>
</tr>
<tr>
<td>SsCOGA</td>
<td>0.811</td>
</tr>
<tr>
<td>mssCOGA</td>
<td>0.911</td>
</tr>
</tbody>
</table>

**Table 6.15:** Relative set cover (miniCAPS is set to maximise ferry range) \( \hat{C}_m = \hat{C}_m^{100} \)

The average FCS fitness \( \langle FCS_{ave} \rangle \) and set cover results from vmCOGA, amCOGA, HiCOGA, SsCOGA and mssCOGA application to miniCAPS, TBCOM and Keane's function are shown in table 6.14. The corresponding relative set cover values (along with those for TF1, TF2 and the 6d-test function) are shown in table 6.15. In general the smaller RSC values noted for SsCOGA and mssCOGA upon application to TF1 and TF2 are not evident in any of the three higher dimensionality functions. Moreover, when applied to miniCAPS, the results indicate considerable improvement. As in earlier studies the results for HiCOGA continue to show little increase in RSC across any of the test functions. A general drop in RSC for all adaptive algorithms when applied to Keane's function is noted, however the SsCOGA results are still higher than the 2d examples and only marginally lower in the case of mssCOGA.

Although the \( \langle FCS_{ave} \rangle \) results are in close agreement in the case of miniCAPS and TBCOM, there are some differences when applied to Keane's function. These increases suggests higher levels of FCS convergence, and may indicate a reduced ability of Halton injection and spatial selection to maintain diversity for prolonged search periods, again further work should address this.
These results of this higher dimensionality study, although of limited scope and comprehensiveness, indicate that the reduction in RSC that caused poor RD performance within the 2d search domains do not occur when applied to functions of higher dimensionality. This coupled with the uniformity (with increasing dimensionality) of both the vmCOGA convergence profiles (figures 6.3-6.6) and the calibration curve (figure 6.15) supports the argument that SsCOGA and mssCOGA performance is maintained under higher dimensionality conditions. Nevertheless, increased FCS convergence noted for both SsCOGA and HiCOGA when applied to Keane’s function is cause for concern and as such necessitates further investigation.

6.7 Concluding Remarks

The adaptive techniques presented in this chapter have eliminated the need to set the Halton injection and spatial selection rates of HiCOGA and SsCOGA respectively. A further technique, mssCOGA is proposed that adapts both spatial selection and mutation, thereby eliminating the need to set a baseline mutation rate for SsCOGA. Adaption is achieved by using a metric that measures the degree and rate of convergence within the population; evidence suggests that this metric is largely independent of the dimensionality of the test function. The desired or target convergence rate is based upon that of vmCOGA, however this changes with increasing dimensionality and search duration and as a result a calibration curve is proposed that allows for application to test functions of higher dimensionality.

Results from studies upon the six dimensional test function of section 4.6 further substantiate the conclusions of chapter five, namely that spatial selection and to a lesser degree, Halton injection may be used to increase the sampling of sensitive regions within the search space. Sampling is not lost if higher convergence rates are used. It is shown that SsCOGA and mssCOGA (using a target convergence rate of one and a half times that
of vmCOGA) outperformed vmCOGA in 94% and 83% of all high performance regions respectively. When applied to lower, 2d test functions, reduced set cover limited RD performance. Further studies indicate that this does not occur upon application to functions of higher dimensionality, however these findings should be corroborated. Nevertheless, it is recommended that in the unlikely event that 2d test functions are to be investigated in any practical context, vmCOGA amCOGA (with a target convergence rate equal to vmCOGA) or the 30x30 ECOGA should be used.

Throughout this and the preceding chapter one of the more significant drawbacks when increasing the sampling of sensitive regions within the search space is found to be a reduction in algorithm robustness. Since COGA is used in an environment where multiple iterative and exploratory runs are executed, consistency in the results is paramount. The following chapter addresses this by introducing a technique that increases the robustness of search by significantly reducing the standard deviation of both set cover and GDE.
CHAPTER 7

The Dynamic Adaptive Filter

7.1 Introduction

The preceding chapters have explored in considerable detail the relationship between the diverse search engine and COGA performance. Although the generation of solutions is crucial and therefore warrants such investigation, discussion relating to the extraction of solutions by the adaptive filter (AF) has not been discussed.

It has been shown that the AF models the fitness distribution of the population by approximating it to the standardised normal distribution (section 2.2.2). The performance of the AF is therefore dependent upon the assumption that the population fitness distribution at any generation is normally distributed and remains so throughout search. In certain search scenarios this assumption may be prone to error, and in extreme cases false. An example being a heavily constrained search space, where the fitness distribution is distorted by the action of the constraint equation(s).

The Dynamic Adaptive Filter (DAF) increases the robustness and accuracy of COGA by reducing the effects of two causes of poor filter performance, model drift and model mismatch. This is achieved by hybridising explicit and implicit filtering into a single operation which occurs at the end of each generation (after the first search stage). When the population is filtered using the DAF, a library of Probability Density Functions (PDFs)(section 7.2.) is used to model the population fitness distribution. The PDF that produces the closest match to the actual fitness distribution, according to a "closeness of fit" metric, is used to model the fitness distribution of the population (section 7.3).
probability mapping then converts the value of $R_f$ to an equivalent fitness threshold ($f_{R_f}$) for the chosen distribution (section 7.5). A population member is then copied to the FCS if the fitness exceeds the $f_{R_f}$ threshold.

7.1.1 Model Drift

Explicit filtering involves the normalisation of the population fitness distribution and the comparison against a predefined filtering threshold (section 2.2.2.3). During intermediate generations a solution is also copied to the FCS if its true fitness exceeds that associated with the previous filtering threshold (implicit filtering) (section 2.2.2.4). Implicit filtering is therefore prone to a degree of inaccuracy since the actual population mean and standard deviation change during the generations between explicit filtering. This phenomenon is referred to as model drift and is eliminated by explicitly filtering the population at the end of each generation (after the first search stage).

7.1.2 Model Mismatch

Model mismatch occurs when the theoretical population distribution does not accurately model the true population. The problem of model mismatch will further augment the inaccuracies caused by model drift. The two scenarios where the actual population distribution may depart from that described by the normal distribution are referred to as kurtosis and skewness.

7.1.2.1 Kurtosis

Kurtosis is a measure of the peaked nature of a distribution and is measured relative to the normal distribution (figure 7.1). Kurtosis may be either leptokurtic (where there are a greater proportion of solutions at the extremes of the distribution) or platykurtic (where there are a greater number of solutions located about the mean of the distribution). If the
degree of kurtosis becomes too great the normal PDF no longer adequately models the true fitness distribution and model mismatch occurs.

![Figure 7.1: Kurtosis of the normal distribution](image1)

![Figure 7.2: Skewness of the normal distribution](image2)

7.1.2.2 Skewness

The degree of symmetry of a distribution is measured by skewness (figure 7.2.). If the distribution has a greater proportion of solutions located at its head (lower quartiles of its range) it is said to be positively skewed. If there is a higher proportion of solutions at the tail, the distribution is negatively skewed. As with kurtosis, if skewness becomes too great, model mismatch may result.

7.2 The Probability Density Function Library

The PDFs within the library are selected to cover the whole envelope of possible population fitness distributions. The following section briefly reviews each PDF, and indicates instances where the true population fitness distribution may closely relate to the distributions given by the library PDFs.

Note that to maintain generality the continuous variable is denoted as $x$, however the terms $x$ and fitness are interchangeable. Furthermore $X$ is a random variable of the range $x_{min} \leq X \leq x_{max}$ and $n$ is the number of observations.
7.2.1 Uniform Distribution

During the initial generations of search, the evolving population may contain diverse solutions that have not converged upon the fitter regions of the search space. As a consequence of random initialisation, the population is also likely to be evenly dispersed throughout the search space and may result in a fitness distribution that is uniform in nature. The uniform distribution has a flat PDF between two bounds, $a$ and $b$. The probability of a random solution having any fitness between the upper and lower bounds is constant across the whole fitness range (figure 7.3).

$$f(x) = \begin{cases} \frac{1}{b-a} & a \leq x \leq b \\ 0 & \text{elsewhere} \end{cases}$$

$$p(X = x) = \begin{cases} 0 & x < a \\ \frac{x-a}{b-a} & a \leq x \leq b \\ 1 & x > b \end{cases}$$

$$\text{mean} = \frac{a+b}{2} \quad \text{variance} = \frac{(b-a)^2}{12}$$

Figure 7.3: The uniform distribution ($a = 0.25$, $b = 1.25$)

7.2.2 Exponential Distribution

In many real world design domains such as TBCOM, models are constrained to avoid operation within undesirable or potentially dangerous regions of the search space. This may result in an exponentially distributed population fitness distribution where large numbers of solutions lie at the lower quartiles of the fitness range, with relatively lower proportions lying at the higher quartiles (figure 7.4).
7.2.3 Normal Distribution

As search begins to converge upon high performance regions, the number of solutions populating infeasible regions decreases and the fitness distribution may assume the "bell shape" of the normal distribution (figure 7.5). For a full description of the normal distribution and the role it plays within the AF, reference should be made to section 2.2.2.2.

\[ f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, \quad -\infty < x < \infty \]

\[ p(X \leq x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-\frac{(t-\mu)^2}{2\sigma^2}} dt \]

\[ \text{mean} = \mu = \frac{\sum x}{n} \]

\[ \text{variance} = \sigma^2 = \frac{\sum (x-\mu)^2}{n-1} \]

Figure 7.5: The normal distribution ($\mu = 0.75$, $\sigma = 0.25$)

N.B. $P(X \leq x)$ is calculated by using Hastings's best approximation (appendix F).

7.2.4 Lognormal Distribution

As search continues to converge upon high performance regions, it is possible that the fitness distribution will not assume the perfectly symmetrical appearance of the normal
distribution but may be positively skewed producing a greater proportion of solutions that lie at the lower quartiles of the fitness range. As the name suggests, the lognormal distribution is an adaption of the normal distribution, allowing for such skewing (figure 7.6).

The application of the lognormal distribution is simplified greatly, by taking logarithms of all points within the sample. In doing this, the data can be treated as normally distributed for the purposes of hypothesising a distribution, parameter estimation and goodness-of-fit testing [Law et al. 1990].

7.2.5 Weibull Distribution

The assumption that the fitness distribution will develop into either a symmetrical or positively skewed bell shape does not account for the possibility of a negatively skewed distribution where there are large number of solutions that lie at higher quartiles of the fitness range. This is more likely to occur at the latter stages of search where the population converges upon optimal areas of the search space. The Weibull distribution may provide an accurate model in such cases since it allows for both positive and negative skewing (figure 7.7).
2.5.---------------.,
2
1.5
0 .5
O·L----~--------~
0
0 .3 0 .6 0 .9 1.2
1.5
X
l
a a - 1 _ ,i
f(x)=
e^\frac{x}{\theta}
0
elsewhere

\( p(X \leq x) = 1 - e^{-\frac{x^\alpha}{\theta}} \quad x > 0 \)

\( \alpha \) and \( \theta \) are calculated directly from the data
(Appendix G).

Figure 7.7: The Weibull distribution \((\alpha = 6.2, \theta = 1.2)\)

7.3 Closeness of Fit Metrics

The closeness of fit or \( d \)-metric is used to determine which of the five theoretical PDFs
gives the closest match to the actual population fitness distribution. If the theoretical
distribution perfectly matches the actual distribution the \( d \)-metric approaches zero, larger
values indicate less accurate modelling. Two closeness of fit metrics are presented, Chi-
squared \((\chi^2)\) and Kolmogorov-Smirnov.

7.3.1 Chi-Squared \((\chi^2)\) Metric

The \( \chi^2 \) test [Kanji 1993] groups the sample distribution into \( k \) discrete classes. The
expected and actual number of points in each class is compared to give an overall \( d \)-metric.
The \( \chi^2 \) metric is determined by the following procedure,

1. Sort sample population into ascending order.
2. Divide the sample into \( k \) classes.
3. Determine the number of points \( n_i \) that lie in each of the \( k \) classes \((1 \leq i \leq k)\).
4. Using the theoretical distribution calculate the probability of falling into each
class \((p_i)\).
5. The \( d \)-statistic is given by,
The main drawback of the $\chi^2$ test is the need to select both the number and boundary values of the discrete groups. The equiprobable approach [Law et. al. 1990] offers a solution by automatically selecting the class boundaries to ensure,

\[
p_i = \frac{1}{k} \quad \forall i
\]

Yarnold [Yarnold, 1970] argues that the equiprobable test is valid only if,

\[
k \geq 3 \quad \text{and} \quad np_i \geq 5 \quad \forall i
\]

The equiprobable $\chi^2$ test used within the DAF has 10 discrete classes. Since $np_i \geq 5$, and assuming $k = 10$, the smallest population size permitted is 50.

### 7.3.2 Kolmogorov-Smirnov Metric

Unlike the $\chi^2$ metric, the Kolmogorov-Smirnov (KS) test does not group the sample distribution into discrete classes [Kanji 1993]. Furthermore, the KS test is valid for any value of $n$ and may be used when $np_i < 5$. The KS metric is determined by the following procedure,

1. Sort the sample population into ascending order ($x_1, x_2, \ldots, x_n$)
2. Assign an ascending rank $(i)$ to each sorted population member (1 ≤ $i$ ≤ $n$).
3. Calculate,

\[
D^+ = \max_{1 \leq i \leq n} \left[ i \cdot P(X \leq x_i) - \frac{i}{n} \right]
\]

\[
D^- = \min_{1 \leq i \leq n} \left[ P(X \leq x_i) - \left( \frac{i-1}{n} \right) \right]
\]

where $X$ is a random variable of the range $x_{min} \leq X \leq x_{max}$.
4. The d-statistic is given by,

\[ d_{\text{statistic}} = \max(D^+, D^-) \]

The application of the KS test has been shown to be more limited than the \( \chi^2 \) test [Conover 1980]. If the KS test is applied to a sample where the distribution parameters have been estimated from the data, the test becomes conservative and may give larger d-statistics values. This phenomenon is less critical in the case of the DAF since the conservatism will be evident (to the same degree) in all cases and will therefore have little effect when a relative comparison is made. Nevertheless, within the DAF the KS test is only used when \( np_i < 5 \).

7.4 Application of PDFs to Multi-Dimensionality Test Functions

To justify the replacement of the AF by the DAF, five tests are executed which indicate search scenarios where the ability of the library PDFs to model the fitness distribution supersede that of the normal PDF. In each example, the d-metric clearly illustrates which PDF gives the closest modelling of the fitness distribution at any particular generation (smaller d-metric values indicate more accurate modelling).

In each test case the d-metric corresponding to each PDF is averaged over 100 independent vmCOGA runs. The generation vector \((g)\) is \([50, 100, 150, 200, 250]\). In the first four examples the mutation vector \((m)\) is \([0.08, 0.06, 0.04, 0.02, 0.01]\). In the final example the effects of differing search scenarios are investigating by using the mutation vector to force differing search profiles, normal search, vmCOGA search, random walk and premature convergence in the same manner as section 6.3. The value of the filtering vector is arbitrary since filtering does not interfere with the evolving population and will therefore have no effect upon the fitness distribution during search.
7.4.1 DeJong F5

Test function five of the DeJong five function test bed [DeJong 1975], also referred to as Shekel's foxholes, is a 2d test function of the form,

\[
f(x_1, x_2) = 0.002 + \sum_{j=1}^{5} \frac{1}{j + \sum_{i=1}^{5} (x_i - a_{ij})^6} \quad -65.536 \leq x_i \leq 65.536 \quad \ldots 6-6
\]

The function consists of multiple, discrete peaks of equal magnitude, and is therefore highly discontinuous.

![Figure 7.8: Application of library PDFs to DeJong's F5 function](image)

Referring to figure 7.8, it is apparent that the lognormal distribution outperforms\(^1\) the normal distribution and every other PDF at every generation up to the 200\(^{th}\) generation where \(p(\text{mutation})\) drops from 0.02 to 0.01. After this point the normal, lognormal, uniform and Weibull distribution all produce near identical d-metrics. An interesting

\[\]

\(^1\) Performance in this context is taken as the closeness of fit (give by the d-metric) between the actual population distribution and the theoretical population distribution obtained from the relevant PDF.
characteristic of this plot is the large drop in all d-metrics in the generations between 150 and 200, this corresponds to a mutation probability of 0.02.

7.4.2 Michalewicz's Function

The Michalewicz and Schoenauer function [Michalewicz et. al. 1996] is a 2d, non-linear function of the form,

\[ f(x_1, x_2) = \frac{\sin^2(2\pi x_1) \sin^2(2\pi x_2)}{x_1(x_1 + x_2)} \quad 0 \leq x_i \leq 10 \]

the function has two hard constraints given by,

\[ x_1^2 - x_2 + 1 \leq 0 \]
\[ 1 - x_1 + (x_2 - 4)^2 \leq 0 \]

The values of the function are truncated at ±10, allowing solutions in the feasible regions to be easily distinguishable. The function is characterised by many local optima and a number of global optima located along the x₁ axis.

![Figure 7.9: Application of library PDFs to Michalewicz's function](image-url)
During the most diverse stage of search \( p(\text{mutation}) = 0.08 \), the Weibull distribution provides the closest model. After the 50\(^{th}\) generation both the Weibull and lognormal distributions begin to diverge and are out performed by all three remaining distributions. Between generations 50 and 100 the normal, exponential and uniform distributions all do equally well. As the mutation probability drops and search begins to converge further, the normal distribution proves to be the most accurate model. A major feature of figure 7.9 is the noticeable jumps in the d-metric curves periodically every 50 generations. These correspond to the drops in mutation probability that occur at the end of each search stage. Interestingly, this investigation is not the first time this phenomenon has been encountered. The convergence rate profiles of figures 6.3-6.6 also show this characteristic response.

### 7.4.3 MiniCAPS (Maximising SEP2)

In this test case, miniCAPS (section 2.3.1) is used to maximise SEP2 (Supersonic Specific Excess Power).

![Figure 7.10: Application of library PDFs to miniCAPS (minimising SEP2)](image)

Immediately noticeable is the stepped nature of figure 7.10, however unlike the Michalewicz function, only the uniform and exponential PDFs exhibit this characteristic.
In further contrast to the other test functions, the uniform distribution produces the most accurate model for all the generations during the first and second search stages. As search starts to converge upon high performance regions, the performance of both the uniform and exponential distributions decrease and are finally superseded by the “bell” type PDFs, with the normal distribution dominating after the 100th generation.

### 7.4.4 Keane’s 20 Dimension Bump Function

To briefly recap, the Bump function is a high dimensional non-linear function with multiple optima, it contains one linear and one non-linear constraint (section 6.3). Added complexity is inherent since the global optimum is defined by the presence of a constraint boundary [Keane 1994].

![Graph](image)

**Figure 7.11:** Application of library PDFs to Keane’s 20d Bump function (uniform and exponential curves omitted)

Since the performance of the exponential and uniform PDFs are comparatively poor (located at d-metric values of 200 and 100 respectively) they are omitted from figure 7.11.

Throughout the 250 generations, the normal distribution is outperformed by both the lognormal and Weibull distributions, suggesting that this function tends to produce higher
numbers of solutions that lie at the lower quartiles of the fitness range (positive skew). As
the mutation probability decreases, the Weibull distribution no longer competes with the
lognormal distribution. This result is interesting since the Weibull distribution can model
both positively and negatively skewed populations, and should therefore compete with the
lognormal distribution in this instance.

7.4.5 TBCOM

The effects of convergence profiles upon the performance of the DAF are investigated by
modifying the mutation probability (section 6.3).

In all four examples, the normal distribution outperforms all remaining PDFs, suggesting
that the TBCOM model produces many solutions that lie about the mean fitness value and
fewer solutions that lie at the extremities of the fitness range. The performance of the remaining four PDFs varies from one profile to another. If the distributions are ranked in terms of performance, the ranking is maintained when looking at the more conservative profiles (normal and vmCOGA search). When looking at more radical search scenarios (random search and premature convergence) the ranking is lost. This result is not surprising since both search profiles are extreme examples, not usually encountered in normal COGA operation. In the case of premature convergence, the population converges upon a single fitness value, resulting in a standard deviation of zero, at this point any attempt to model the population using a PDF will fail. With random search, high mutation ensures that there is little convergence between the start and end of search, resulting in the constant nature of figure 7.15. The only perturbations in the curves being attributable to noise induced by the higher mutation probabilities.

In the majority of cases, the performance of the d-metric appears to be independent of the search profile and dependent only upon the test function. If the DeJong F5 and Michalewicz's functions are taken as examples it can be seen that in the former function the Weibull distribution performs relatively poorly for higher mutation values, whilst with the latter function the Weibull distribution performs well at high mutation probabilities. A correlation between d-metric and mutation probability is however noted in figures 7.9, and in some of the curves of figures 7.10 and 7.12, where the stepped nature of the plots relates to the changes in mutation probability at the end of each search stage.

The normal distribution has been shown to perform adequately well over the whole range of functions and therefore justifies its inclusion as the single PDF used within the original AF. However in many cases the normal distribution is outperformed by one or more of the remaining PDFs.
The uniform and exponential distributions perform poorly over the majority of functions. Nevertheless, it is unwise to assume that this will be true in all future cases (generations 0 to 100 of figure 7.10, suggests that this is certainly the case for the uniform PDF), both distributions therefore remain in the DAF.

7.5 Filtering Threshold Probability Mapping

The filtering threshold value ($R_f$) is based upon application to the standardised normal distribution (section 2.2.2). To maintain consistency across the library PDFs, a mapping must be made to determine the equivalent $R_f$ value when a library PDF is selected for the filtering process.

Consider a standardised normal fitness distribution, for a given filtering threshold $R_f$, the probability of a random solution with standardised fitness $f$, passing the filtering process ($P(\text{Pass}_{R_f})$) and entering into the FCS is indicated by the shaded regions of figure 7.16 and given by,

$$P(\text{Pass}_{R_f}) = 1 - \int_{-\infty}^{R_f} \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(f-\mu)^2}{2\sigma^2}}$$

...6-10

The equation is reduced since $\mu = 0$ and $\sigma = 1$.

$$P(\text{Pass}_{R_f}) = 1 - \int_{-\infty}^{R_f} \frac{1}{\sqrt{2\pi}} e^{-\frac{R_f^2}{2}}$$

...6-11

The equivalent fitness ($f_{R_f}$) for any PDF (figure 7.17) which gives a probability of passing filtering of $P(\text{Pass}_{R_f})$ may be obtained by rearranging,

$$1 - P(f_i \leq f_{R_f}) = P(\text{Pass}_{R_f})$$

...6-12

Where $P(f_i \leq f_{R_f})$ for each PDF is given in figures 7.3 to 7.7.
Consequently, a solution is copied to the FCS if its true fitness exceeds the value of $f_{RF}$. This mapping ensures that approximately similar (allowing for stochastic effects) proportions of solutions are copied to the FCS for any PDF and for any given value of $R_f$.

### 7.6 Assessing the Performance of the DAF

To assess the performance of the DAF, vmCOGA algorithms using both the AF and the DAF filter are applied to the 6d-test function. As with all investigations, the results were averaged over 100 independent trials. To maintain consistency, the mutation, generation and filtering vectors are set to values used in section 4.7.

$$m = \{0.08, 0.06, 0.04, 0.02, 0.01\}$$

$$g = \{25, 50, 75, 100, 125\}$$

$$R_f = \{1.75, 1.75, 1.75, 1.75, 1.75\}$$

The effects of eliminating model drift can be seen by comparing the results of the control filter and filter A. The control filter is the AF introduced in section 2.2.2, and uses both explicit and implicit filtering. Filter A is a standard AF with implicit filtering replaced by explicit filtering at every generation (excluding the first search stage), effectively isolating the effects of model drift. Reduced set cover, increased GDE and increased average FCS fitness, all indicate a higher levels of convergence present in the FCS when model drift is
eliminated (table 7.1). When looking at regional decomposition (figure 7.18), it can be seen that smaller set cover and higher GDE combine to produce fewer results in all regions when compared with the AF control, however this decrease in RD must be offset by a dramatic decrease in both set cover and GDE sensitivity (standard deviation).

Model drift is eliminated and model mismatch reduced by using the DAF (filter B). The degree of convergence within the FCS reduces when compared with filter A, indicating slightly less discriminatory filtering, generally increasing the overall number of regional hits. Furthermore, the set cover and GDE sensitivities have decreased and increased respectively. Both sensitivities are again considerably smaller than those associated with the control filter.

It may be concluded that reducing model drift and model mismatch have the side effect of producing slightly more discriminatory filtering (increased GDE and reduced set cover). It is suggested that when using the DAF, the filtering vector ($R_f$) may be reduced slightly to produce comparable decomposition results whilst still maintaining the large reduction in both set cover and GDE sensitivity. Considerable empirical effort has indicated that there is no single reduction factor that yields an exact match for the set cover and GDE results of the control filter, however if these two measures are considered independently, comparable results are achieved.

<table>
<thead>
<tr>
<th>Filter</th>
<th>Filter</th>
<th>Set Cover</th>
<th>GDE</th>
<th>FCS$_{set}$</th>
<th>FCS$_{best}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control</td>
<td>AF (Explicit and Implicit Filtering)</td>
<td>285.72</td>
<td>42.11%</td>
<td>1.344</td>
<td>1.471</td>
</tr>
<tr>
<td></td>
<td>$R_f = (1.75, 1.75, 1.75, 1.75, 1.75)$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>AF (Explicit Filling)</td>
<td>178.06</td>
<td>51.35%</td>
<td>1.354</td>
<td>1.471</td>
</tr>
<tr>
<td></td>
<td>$R_f = (1.75, 1.75, 1.75, 1.75, 1.75) $</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>DAF</td>
<td>208.64</td>
<td>47.75%</td>
<td>1.349</td>
<td>1.472</td>
</tr>
<tr>
<td></td>
<td>$R_f = (1.75, 1.75, 1.75, 1.75, 1.75) $</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>DAF (Match GDE)</td>
<td>248.48</td>
<td>41.99%</td>
<td>1.341</td>
<td>1.471</td>
</tr>
<tr>
<td></td>
<td>$R_f = (1.70, 1.70, 1.70, 1.70, 1.70) $</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>DAF (Match Set Cover)</td>
<td>285.98</td>
<td>38.44%</td>
<td>1.337</td>
<td>1.469</td>
</tr>
<tr>
<td></td>
<td>$R_f = (1.65, 1.65, 1.65, 1.65, 1.65) $</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7.1: FCS results for vmCOGA using the DAF (values in parenthesis are standard deviations expressed as a % of measured value)
Filter examples C and D are obtained by relaxing $R_f$ to a point where the GDE (filter C) and the set cover (filter D) approximately match the corresponding values for the control filter. However, the attainment of these values results in a much lower value for the remaining measure. If filter D is taken as an example, $R_f = \{1.65, 1.65, 1.65, 1.65, 1.65\}$ results in approximately equal set cover but at the expense of a reduction in the corresponding GDE, this is also noted with filter C. Although the RD of both filters is still lower than those associated with the control filter, they offer notable improvement over both filter A and B, and produce similar performance in terms of both set cover and GDE sensitivity.

7.7 Integrating the DAF with SsCOGA and mssCOGA

If one applies the DAF to the more successful adaptive algorithms of chapter 6 (SsCOGA and mssCOGA using a target convergence rate of $1.5\hat{C}_w^m$), the general increase in FCS convergence noted in the preceding section is again evident in rows four and seven of table 7.2 (reduced set cover and increased GDE).
Table 7.2: FCS results for SsCOGA and mssCOGA using the DAF ($\hat{c}''_n = 1.5\hat{c}'_n$) (values in parenthesis are standard deviations expressed as a % of measured value)

A more accurate comparison is once again made by relaxing the filtering vectors to achieve GDE values that are comparable with the SsCOGA and mssCOGA AF algorithms. The corresponding $R_f$ vectors are $\{1.55, 1.55, 1.55, 1.55, 1.55\}$ and $\{1.65, 1.65, 1.65, 1.65, 1.65\}$ for SsCOGA and mssCOGA respectively. The resulting reduction in the standard deviation of set cover and GDE in rows five and nine of table 7.2, indicates significant increases in algorithm robustness when compared with the adaptive algorithms of chapter 6 and vmCOGA.

Figure 7.19: Regional decomposition results for SsCOGA using the DAF and the relaxed filtering vector
In this instance, there is little evidence to suggest that the DAF has any significant effect upon overall RD performance. As expected, there are regional differences in RD (figures 7.19 and 7.20). However, such differences appear to balance out across groups and average performance rank is not affected in four of the six sensitivity groups (table 7.3). The exceptions to this being the low sensitivity group of SsCOGA and medium sensitivity group of msSCoGA respectively, in the case of SsCOGA the difference in performance rank indicates superior performance in the case of the DAF.

<table>
<thead>
<tr>
<th>Number of regions where vmtCOGA is outperformed</th>
<th>Average performance rank (1-2) when compared with corresponding adaptive algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>SsCOGA (AF - C6)</td>
<td>High</td>
</tr>
<tr>
<td>6/6</td>
<td>1.50</td>
</tr>
<tr>
<td>SsCOGA (DAF)</td>
<td>6/6</td>
</tr>
<tr>
<td>msCOGA (AF-C6)</td>
<td>4/6</td>
</tr>
<tr>
<td>msCOGA (DAF)</td>
<td>4/6</td>
</tr>
</tbody>
</table>

Table 7.3: RD results for SsCOGA and msCOGA (relaxed filtering vector) using the DAF (Low, medium and high relate to sensitivity groupings)
7.8 Concluding Remarks

This chapter has addressed the role the adaptive filter plays in the overall performance of COGA. The standard deviation of both set cover and GDE has been greatly reduced by removing implicit filtering and explicitly filtering every generation after the first search stage. This increased accuracy has the side effect of producing characteristically more discriminatory filtering which reduces set cover and increases GDE. It is shown that this may be alleviated by reducing the effect of model mismatch, and is achieved by replacing the Adaptive Filter with the Dynamic Adaptive Filter. Empirical evidence suggests that the decrease in set cover and increase in GDE may be further reduced by relaxing the value of the filtering vector, with little negative effects upon the overall robustness.
8.1 Introduction

The penultimate chapter of this thesis deals with the final operation block of the COGA mechanism, entitled "knowledge discovery" (figure 2.1). In an engineering design sense, the importance of this block can not be overstated, since it provides the link between the designer and the design domain under investigation and is therefore the only means of knowledge discovery within COGA.

Knowledge discovery (KD) and more specifically knowledge discovery in databases (KDD) is defined as being the extraction of implicit, previously unknown and potentially useful information (or patterns) from data [Frawley et. al. 1992]. In terms of COGA, the FCS represents the database and the information relates to relationships that are imbedded within the data. If miniCAPS is used as an example, a hypothetical relationship may be "there are no solutions within the FCS that have aspect ratios greater than 3.0 and gross wing plan areas of less than $67\text{m}^2$". However during any KD process it is also important to balance the amount of presented information against the ability of the human mind to efficiently interpret it, if this becomes too great, information overload occurs where the designer is unable to process all the presented data and the quality and degree of overall knowledge attained reduces [Noyes 1995].

It is not the intention of this chapter to discuss the many techniques to aid knowledge discovery, but to introduce the most suitable for use with COGA and to give practical examples of their application to miniCAPS and TBCOM where appropriate.
As with all KD techniques, the data set under investigation should be as large as possible and of sufficient quality to increase the accuracy and reliability of the results. Consequently, the values within the filtering vector are set to characteristically small values (1.0 or 1.5) to promote the generation of large FCS sizes that reduce the effects of sampling error.

8.2 Visualisation

Visualisation is the simplest and most powerful method of analysing data [Cleveland 1993]. It differs from the more involved data mining techniques such as decision trees, clustering, classification etc. since it uses the human to interpret the presented information and to extract the required knowledge, it is argued that the performance of this approach more often exceeds datamining [Cox et. al. 1997]. This coupled with its comparable simplicity makes visualisation the most widely used knowledge discovery technique. It should be noted that the graphs and hyperplane plots used throughout this thesis are all examples of knowledge discovery through visualisation.

Over recent years, research relating to the visualisation of GAs has become an increasingly active area. There are two general branches; the first deals with visualising the state and course of a GA run [Pohlheim 1999]. This involves the presentation of data such as optimal population fitness and the corresponding design variables taken at a particular generation. The course of a GA is presented by plotting the aforementioned data over the duration of a GA run. It is clear that with respect to the COGA environment, such methods are unsuitable because they give little information relating to the geographic nature of the solutions. The second method, is more closely related to the aims of this chapter and involves the mapping of higher dimensionality data into two or three-dimensional representation [Shine et. al. 1997, Collins 1997, 1999]. However in doing this, reference to real world parameters such as velocity, pressure, temperature etc. may be lost. The COGA
user may also not be familiar with the mathematically complex transformation methods employed, hence the ability to rapidly assess the interaction between design variables without appropriate training is questionable. Such mapping techniques are therefore not discussed further in this chapter.

The first question one must ask when visualising the FCS is what data to present and how it should be presented 1-dimensional, 2-dimensional etc. Throughout this chapter, the data within the FCS is split into two classes, design variable and design attribute data. Design variable data is the actual values of the data stored in the FCS that collectively define each design solution, i.e. aspect ratio, wing area, cruise Mach number etc. Design attribute data defines some attribute of the individual design solution, fitness, constraint violation, unit cost and so on. The dimensionality of a visualisation technique is taken as the number of design variables that are presented simultaneously. For instance, a visualisation technique showing wing sweep, climb Mach number and fitness is deemed to be a 2-dimensional approach, which presents an additional design attribute.

As the dimensionality of the applied visualisation technique increases, the number of possible design variable combinations that may be presented also increases (table 8.1). The relationship between the number of possible plots and the number of presented design variables is given by,

\[ \text{number of possible design variable combinations} = \frac{n!}{v_r!(n-v_r)!} \]  

where,

\[ n = \text{total number of design variables} \]
\[ v_r = \text{number of presented design variables} \]
<table>
<thead>
<tr>
<th>Presented design variables (n)</th>
<th>Number of possible design variable combinations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1d visualisation</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 8.1: Number of design variable combinations

The least graphically demanding system of representation is one-dimensional plotting where each variable is examined individually. The following section discusses three specific examples, bar, density and box plots.

8.2.1 One Dimensional (Univariate) Visualisation

Although simplistic, 1-dimensional visualisation is surprisingly powerful since it allows for a rapid assessment of the general distribution of each design variable across its operating range. Since COGA search converges upon regions of high performance, it is assumed that areas of increased solution density relate to regions of high performance.

The distribution of solutions across the range of a design variable therefore gives an indication of the degree of interaction between the design variable and the fitness of the optimised objective; this is termed design variable sensitivity (DVS). Low DVS indicates that any given value of a particular design variable (within its range) has little effect upon the overall fitness of the design solution.

Figure 8.1 presents bar plots of two of the eight miniCAPS design variables when ferry range is maximised. In each example the range of each variable is discretised into any number of class intervals (in this instance 20). The number of solutions, average fitness and standard deviation of fitness for the solutions in each class is presented.
The more uniform nature of figure 8.1 suggests that there is no region or regions within the operating range of taper ratio that produces high performance (low DVS). In the case of wing area there is a clear region containing all the solutions within the FCS (relating to a wing area of greater than 62m² and indicating a high DVS). In may be concluded that a solution with a wing area of less than 62m² will not produce a solution of high fitness.

Figures 8.3 and 8.8 give a measure of the change in design solution fitness with small perturbations to the design variable. Across the range of taper ratio it can be seen that there are large changes in fitness sensitivity, whilst in the high performance region of wing area, the fitness sensitivity is generally smaller and more uniform.

Box plots are used to summarise the scale, shape and distribution of the data set. They are useful since multiple data sets can be compared simultaneously and without the need for
discretisation. If the sample data from the FCS containing \( n \) samples of a given design variable or attribute is ranked in ascending order, the lower (LQ) and upper quartiles (UQ) are the \( 0.25(n+1) \)th and \( 0.75(n+1) \)th data sample respectively. These quartiles are represented by the two ends of the box in figure 8.7. The distance between these points is referred to as the interquartile range (IQR) and is a measure of the spread of the distribution. The line inside the box corresponds to the median (middle) value of the distribution and gives a measure of the location of the distribution, it may be used with the upper and lower quartiles to determine the skewness of the distribution. The upper adjacent (UA) value is the greatest value that lies within a point given by UQ+1.5×IQR. Similarly, the lower adjacent (LA) value is the smallest value that lies within LQ-1.5×IQR. UA and LA values are used to measure the spread and shape in the extremes of the distribution. Values that lie outside of UQ+1.5×IQR and LQ-1.5×IQR are characterised as outliers.

Figure 8.7: A sample box plot

Figure 8.8: Box plot of fitness relating to each geometry configuration

Figure 8.8 shows box plots representing the fitness distributions for each discrete geometry configuration within a COGA run upon TBCOM. The box plot relating to geometry one, indicates considerably lower performance when compared with the remaining geometries. Furthermore the distribution has a very small spread and appears to be positively skewed.
towards the lower regions of its distribution (further justification for the use of the DAF
due to the non-normality of the distribution). The results for geometry two and three
indicate considerably higher performance and fitness spread. Moreover the fitness
distribution in the case of geometry three is approximately symmetric about the median
and can therefore be assumed to be similar to the normal distribution. The presence of an
outlier in this distribution represents an isolated individual with a fitness that is markedly
higher than the general distribution.

Figures 8.9 and 8.10 are box plots illustrating the distribution of solutions within the FCS
presented in terms of Ff and Rp. Although TBCOM is a mixed parameter model, the
ranges of the presented variables are independent of this. Figure 8.9 supports this,
however Rs has three differing ranges each dependent upon geometry. It may be
concluded that with respect to geometry Ff has a low and RS a higher DVS.

The distribution of solutions across the operating range of a design variable may also be
presented in the form of a non-parametric probability density function. This generates the
PDF of the data without the assumption that the data fits some general distribution, normal,
uniform etc. This technique has notable advantages over other 1-d approaches: firstly its is
continuous and does not present the data within discrete intervals; secondly it does not reduce the amount of presented data as is the case with box plots and provides a more efficient medium to compare distributions [Cleveland 1994]. The PDFs for wing sweep when ATR1, STR1 and Take-off mass are optimised are given in figure 8.11. Surprisingly for two apparently related objectives, the PDFs conflict for attained and sustained subsonic turn rate. The former example suggests high performance occurs at higher wing sweep angles. Whilst in the latter case positive skew indicates higher solution density and greater performance at lower wing sweep. When take-off mass is optimised the wing sweep density distribution becomes more uniform, albeit with a slight bias towards lower wing sweep. As with the other 1d approaches, non-parametric PDFs give a measure of DVS. In the case of figure 8.11, DVS for wing sweep relative to ATR1 and STR1 are higher than sweep relative to take-off mass.

![Figure 8.11: Non-parametric PDF for wing sweep when ATR1, STR1 and take-off mass are optimised.](image)

One-dimensional visualisation, although simple and effective, does suffer from a number of drawbacks, the most significant of which is the inability to investigate the interaction between variables. More powerful approaches overcome this by presenting the data under higher dimensionality conditions. It is therefore suggested that 1-d investigation should be
used as a precursor to the more involved analysis methods detailed in the following sections.

8.2.2 Two Dimensional (Bivariate) Visualisation

Two-dimensional visualisation allows for the investigation of interactions between pairs of variables. Nevertheless, a drawback lies in choosing which variables to present since there is a significant increase in the number of possible plots (table 8.1). To overcome this, every possible design variable combination may be plotted within a scatterplot matrix [Cleveland 1993]. An example scatterplot matrix is shown in figure 8.12.

![Scatterplot matrix](image)

**Figure 8.12:** Scatterplot matrix (TBCOM - Geometry 2)

Although exact in the sense that every design variable combination is presented this “brute force” method has many disadvantages. Firstly the overall plot is very crowded and may present too much information. Secondly, this technique also suffers for redundancy with many combinations represented twice (i.e. Rs against Rp is plotted along with Rp against
Finally, more subtle relationships are less apparent and may easily be overlooked. It is clear that as dimensionality increases this technique becomes progressively more cumbersome and alternatives must be sought. A logical choice would be to firstly investigate the variables that have the highest design variable sensitivity. DVS as defined in section 8.2.1 is a qualitative measure, this makes the selection of such variables highly subjective. A solution is to calculate the degree of correlation \( (r) \) between each design variable and fitness since this indicates the nature (positive or negative) and strength of the relationship between the two parameters, and gives a quantitative measure of DVS \( (r_{DVS}) \).

Since the fitness distribution of a COGA population is often not normally distributed, Spearman’s rank correlation is used which overcomes the assumption of normality [Hayslett 1981]. The correlation \( (r_{DVS}) \) between a given number \( (n) \) of examples of design variable \( x \) and the corresponding design solution fitness \( f \) is given by,

\[
r_{DVS} = 1 - \frac{6\sum (X_i - F_i)^2}{n(n^2 - 1)}
\]

Where,

- \( X_i \) Rank of \( i \)th design variable \( (1 \leq X_i \leq n) \)
- \( F_i \) Rank of \( i \)th corresponding design solution fitness \( (1 \leq F_i \leq n) \)

Table 8.2 gives the results of a correlation analysis against fitness, upon the TBCOM model for each design variable within the FCS.

<table>
<thead>
<tr>
<th>Variable</th>
<th>( r_{DVS} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fbc</td>
<td>0.422</td>
</tr>
<tr>
<td>Rp</td>
<td>-0.330</td>
</tr>
<tr>
<td>dth</td>
<td>-0.173</td>
</tr>
<tr>
<td>Rpf</td>
<td>-0.172</td>
</tr>
<tr>
<td>Rs</td>
<td>-0.146</td>
</tr>
<tr>
<td>kw</td>
<td>0.136</td>
</tr>
<tr>
<td>df</td>
<td>0.062</td>
</tr>
<tr>
<td>Cdf</td>
<td>0.021</td>
</tr>
<tr>
<td>Tcl</td>
<td>0.016</td>
</tr>
<tr>
<td>Ff</td>
<td>0.013</td>
</tr>
<tr>
<td>Cdr</td>
<td>0.000</td>
</tr>
</tbody>
</table>

Table 8.2: Correlation analysis for TBCOM Geometry I (Results are ranked in descending absolute magnitude)
If the two highest correlating design variables (Fhc and Rp) are plotted in a 2-dimensional hyperplane (colour of solution represents fitness) a region of high performance in the bottom right quarter is evident (figure 8.13). Fitness generally increases with increasing Fhc (positive correlation) but decreases with increasing Rp (negative correlation). When the two lowest correlating variables (Cdr and Ff) are plotted, there is no specific range of values for either variable that promotes optimal performance. A further point of interest is that the two variables Fhc and Cdr which are dependent upon the discrete variable (geometry) show very differing correlation results (Fhc gives a very strong correlation whilst Cdr gives a very weak correlation).

![Figure 8.13: Design variables with a high correlation](image1)

![Figure 8.14: Design variables with a low correlation](image2)

Similar correlation analysis upon miniCAPS where range, SEPl and ATRl are optimised is given in table 8.3.

<table>
<thead>
<tr>
<th>Maximising Range variable</th>
<th>$r_{DYS}$</th>
<th>Maximising SEPl Variable</th>
<th>$r_{DYS}$</th>
<th>Maximising ATR1 variable</th>
<th>$r_{DYS}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wing sweep</td>
<td>-0.309</td>
<td>Wing area</td>
<td>-0.514</td>
<td>t/c ratio</td>
<td>-0.464</td>
</tr>
<tr>
<td>Wing area</td>
<td>0.257</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cruise height</td>
<td>0.233</td>
<td>t/c ratio</td>
<td>-0.424</td>
<td>Wing sweep</td>
<td>0.422</td>
</tr>
<tr>
<td>t/c ratio</td>
<td>0.225</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aspect ratio</td>
<td>0.191</td>
<td>Cruise height</td>
<td>-0.059</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Climb Mach no</td>
<td>0.121</td>
<td>Taper ratio</td>
<td>-0.031</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Taper ratio</td>
<td>-0.104</td>
<td>Aspect ratio</td>
<td>0.018</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cruise Mach no</td>
<td>0.042</td>
<td>Cruise Mach no</td>
<td>0.014</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 8.3: Correlation coefficient analysis for miniCAPS – maximising Range, SEPl and ATR1 (results are ranked in descending absolute magnitude)
The results of table 8.3 indicate a great deal of variation in \( r_{\text{DVS}} \) when the optimised objective is changed. Wing area, for example, shows positive correlation when both range and \( \text{SEPl} \) are optimised but a negative correlation when \( \text{SEP1} \) is maximised. A further example is wing sweep, which is negatively correlated for range and \( \text{SEPl} \) but positively correlated for \( \text{ATR1} \). Other variables, such as aspect ratio show high correlation for some objectives (\( \text{ATR1} \)) and little in others (\( \text{SEP1} \)). Finally, variables such as cruise Mach number show little correlation for all three objectives.

8.2.2.1 Visualising Design Attributes

Once the design variables of interest have been selected, the next stage is to determine an appropriate visualisation method. There are many design attributes that may be plotted. Figures 8.15 to 8.21 give five methods of representing two-design variables with a further design attribute. The use of colour within a 2d scatterplot is a common approach and has been employed throughout this thesis. Colour may be used to represent categorical data (figure 3.5), a continuous design attribute such as fitness (figure 8.15), constraint violation (section 3.5) or a non-optimised objective (figures 8.16 and 8.17). Presenting a non-optimised objective is useful since it gives a indication (albeit it limited, due to the lack of explicit optimisation) of possible regions of multi-objective compromise, and in this sense may offer an initial approach to multi-objective decomposition. Nonetheless, a drawback of using colour in such a way is that high fitness solutions may be obscured by solutions of lower fitness that lie in the same position. This is referred to as overplotting or occlusion and may be overcome by plotting the points in increasing magnitude of fitness. A further decision must be made as to what colour scheme to use, this may be a single colour (figure 8.22), dual colour (figure 8.24) or a range of colours such as the rainbow spectrum (figure 8.15). Choosing the optimal colour scheme is an area for much debate, the more interested reader should refer to [Tufte 1990 & Travis 1991].
Design attributes may also be represented by the size of each point, where size increases in proportion to the design attribute (figure 8.18). However in densely populated hyperplanes it quickly becomes difficult to differentiate solutions and attribute values, making this technique more difficult to interpret when compared with colour representation.

The remaining three techniques all interpolate or render the points within the FCS to produce a matrix of points (design variable 1, design variable 2 and design attribute) covering the entire hyperplane. There are many interpolation algorithms, several of which are discussed in [Cleveland 1993]. In figure 8.19, the surface is represented as a number of
interpolated design attribute point heights connected by a wire frame. Surface plots suffer from parallax error since the plot is projected into a hypercube topology, this is reduced by rotating the surface and visualising it from a number of differing angles (figures 8.25-8.28). The contour plot consists of a number of isobars or contours, each contour represents areas of equal design attribute magnitude. However, if the plot is noisy, as is the case with figure 8.20, the regions of high performance are difficult to interpret. This may be overcome by superimposing the respective design attribute values upon each contour, nonetheless care must be exercised to avoid cluttering the plot.

Figure 8.19: Design attribute (fitness) represented by surface plot

Figure 8.20: Design attribute (fitness) represented by contour plot

The colourmap is a development of the contour plot and uses colour to represent regions of similar design attribute value (figure 8.21). As in figure 8.15, the colourmap clearly represents the regions of high performance. However, as with the two other rendering techniques, a disadvantage stems from the interpolation technique, since this may introduce false peaks and troughs. Moreover, in sparsely populated regions interpolation becomes increasingly less representative, examples being the lower right hand region of figures 8.19, 8.20 and 8.21.
8.2.3 Three-Dimensional (Trivariate) Visualisation

Visualising three design variables simultaneously can give greater information when compared with the two dimensional methods of the previous section, however choosing the appropriate technique becomes of increased importance, since there is an increased risk of presenting plots that are indecipherable. This scenario is evident when three design variables are presented within a single hyperplane. Figure 8.22 presents the results of a miniCAPS run when ATR1 is maximised. The three highest DVS ranked variables of table 8.3 are presented, t / c ratio and wing sweep are plotted upon the x and y-axis respectively and the colour (single colour representation) depicts aspect ratio. Figure 8.23 presents increased information by using point size and colour to represent aspect ratio and fitness (ATR1) respectively. Although the relationship between the variables plotted upon the axes is immediately evident, the effect of aspect ratio is less forthcoming. It is also difficult to discriminate between dissimilar aspect ratio values within densely populated regions within the hyperplane. When using colour to depict a design attribute such as fitness, overplotting is surmountable by ensuring that the solutions of most interest (i.e. high fitness) are plotted last (figure 8.15). However it is clear that this assumption can not be made when colour relates to design variable value.
A point of further note in figures 8.22 and 8.23 is the overall similarity to figures 3.1 to 3.4, (where landing speed is minimised). Such similarities suggest that for $t/c$ ratio and wing sweep the objectives of maximising ATR1 and minimising landing speed closely concur.

A more powerful trivariate approach is conditioning. Conditioning or coplots illustrate the relationship between two variables with increasing value of the third, conditioned variable [Cleveland 1993]. Figure 8.24 conditions the data presented in figures 8.22 and 8.23 against aspect ratio, two-colour point representation symbolises design solution fitness. Each of the four panels present design solutions that have aspect ratio values falling within the discrete ranges, 1.5-2.5, 2.5-3.5, 3.5-4.5 and 4.5-5.6 (the interval 5.6-6.0 is omitted since no solutions lie within the region). Coplots may be interpreted as visualising the $t/c$, wing sweep hyperplane whilst traversing along the aspect ratio axis.
Within the lower left hand quarter of figure 8.24 (relating to the lowest aspect ratio interval) there is a region of high performance relating to high wing sweep and low $t/c$ ratio. With increasing aspect ratio this region diminishes in size. As aspect ratio increases further the high performance region disappears, also the ridge of lower fitness solutions that arcs from the top left to the bottom right splits and branches into two lower performance regions in the top left and lower right corners of the hyperplane. It follows that coplots, although presenting the same amount of information as figures 8.22 and 8.23, give a much clearer indication of the interactions between three design variables. Furthermore if the number of conditioned widows is made small enough, a dynamic plot can be produced where each conditioned widow constitutes an individual frame of a "movie" [Shine et. al. 1997].
The final trivariate approach discussed here is representation within a hypercube. This approach is arguably the most natural way to visualise three-dimensional data since each design variable is represented upon one of three orthogonal axes. Figure 8.25 presents the data of figures 8.22, 8.23 and 8.24 within a hypercube, this clearly indicates a single region of high performance when aspect and t/c ratio are at the minimum, and wing sweep at the maximum of their respective operating ranges. As with surface plots, a disadvantage of this approach is visual inaccuracy due to parallax. Again this may be resolved by the use of rotation, ideally this should be interactive, but rotation about the aspect ratio axis in steps of 90° provides an acceptable solution (figures 8.26 – 8.28).

Figure 8.25: Thickness / chord ratio, wing sweep and aspect ratio represented within a hypercube, point colour represents design attribute (fitness) — spectral colour representation

Figure 8.26: Hypercube rotated 90° about Aspect ratio axis

Figure 8.27: Hypercube rotated 180° about Aspect ratio axis

Figure 8.28: Hypercube rotated 270° about Aspect ratio axis
8.2.4 Hypervariate and Advanced Visualisation Techniques

Using the techniques detailed so far, a maximum of six design variables may be presented simultaneously: three presented within a hypercube, two represented by point size and colour and then conditioned against the sixth design variable. However, the practicality of any such technique is highly questionable. Nevertheless, relatively clear visualisation of four design variables may be achieved by conditioning a hypercube (figure 8.29).

Figure 8.29: Wing sweep, wing area and thickness / chord ratio represented within a hypercube and conditioned against cruise height, point colour represents design attribute (fitness) - two colour representation

In this example it may be seen that when ferry range is maximised, a single region of high performance is noted that corresponds to high t/c ratios and wing areas and low wing sweep (the design variables are again selected in accordance with table 8.3). A further point of note is that the most pronounced region of high performance appears to fall within the third quartile of the conditioned variables (cruise height).
Further techniques for multivariate data visualisation include linked brushing, icon based and hierarchical methods. Linked brushing, although not a multivariate approach in the strictest sense, is a powerful tool since it allows the user to interactively explore specific regions in terms of any number of design variables. Furthermore brushing may be applied to any representation technique such as 2d, scatterplot matrix, 3d, conditioning etc. Figures 8.30 and 8.31 illustrate how linked brushing may be used to view the data presented in figure 8.29. The user-defined dashed box of figure 8.30 defines the brushed solutions within the t/c ratio vs. wing sweep hyperplane. The highlighted solutions in figure 8.31 are the same brushed solutions presented within the wing area vs. cruise height hyperplane. Although of limited comprehensiveness these results suggest that solutions with low t/c ratios and wing sweep generally have high wing areas and cruise heights.

![Figure 8.30: Linked hyperplane with brushed points (wing area vs. cruise height)](image1)

![Figure 8.31: Linked hyperplane with brushed points (t/c ratio vs. wing sweep)](image2)

Icon based approaches represent the data as features of icons, the most common example of which is a Chernoff face [Chernoff 1973]. Chernoff faces are cartoon-like representations of a face, where each feature depicts the value of a variable. The power of this approach lies in the ability of the human mind to recognise and discriminate between many facial characteristics simultaneously. The Chernoff faces of Figures 8.32-8.34 represent a six dimensional solution at the minimum, medium and maximum of its design
variable values. The variables are represented by: face eccentricity, eyebrow angle, eye size, pupil size, nose size and mouth curvature, however in practice many more variables may be presented.

**Figure 8.32: Chernoff face representing six design variables all at their minimum value**

**Figure 8.33: Chernoff face representing six design variables all at their mid-point value**

**Figure 8.34: Chernoff face representing six design variables all at their maximum value**

Added flexibility may be achieved by presenting Chernoff faces within scatter plots, where two or three (usually of the most important) design variables are presented upon the axis and the remaining variables are represented by the face, once again icon colour may be used to depict a desired design attribute.

Hierarchical methods partition the dataset into lower dimensionality subspaces. In the n-vision or “worlds-within-worlds” approach [Feiner et al. 1990], n-dimensional space is partitioned into nested three-dimensional sub-spaces. For instance, a six-dimensional dataset is represented by having a new co-ordinate system for the last three variables that sits inside the co-ordinate system for the first three design variables. Figure 8.35 consists of two reference systems, the first contains wing sweep, t / c ratio and aspect ratio (red hyperplane), the second corresponds to wing area, taper ratio and cruise height. The value of each design variable is obtained by orthogonally projecting a line to each axis within the corresponding reference frame (represented by the dashed arrow).
A drawback with both icon based and world-within-world approaches is again occlusion, this is augmented when the datasets are sizeable and or the model under investigation highly dimensional. In the hypercube plots of figure 8.25 occlusion is reduced by rotating the plots (figures 8.26-8.28) or by changing the viewing angle. The most efficient means of achieving this is to use an interactive dynamic approach, where multiple plots may be animated and explored within the visualisation domain [Buja et. al. 1996]. This allows the user to "walk around" and zoom into or expand specific areas of interest. As a particular area is expanded, occlusion reduces and the visualisation method may be changed to present increased design information relating to each solution. However, it is still questioned whether such approaches completely eliminate occlusion [Carpendale et. al. 1997]. As a result, recent work has suggested a number of distortion techniques that increase the clarity of specific regions by compressing or distorting the remaining areas of the plot in proportion to their distance from the region of interest (figures 8.36). The overall effect is a decrease in solution density about the focal point without the disruption
of the spatial relationships inherent within the plot, thus allowing for the clearer visualisation of solutions that may be obscured or buried by regions of lesser importance.

**Figure 8.36:** Distortion applied progressively to a hypercube, each individual cube represents an equal sample of the space in the original undistorted configuration. The focal point is located at the centroid (reproduced with permission S. Carpendale © 1997 IEEE)

It follows that such a dynamic environment where the visualisation space may be interactively explored, expanded upon and distorted is practically indistinguishable from non-immersive\(^1\) virtual reality (VR) techniques. Consequently the integration of VR, both immersive and non-immersive, into any prospective visualisation tool should be deemed as a foreseeable possibility.

### 8.3 Concluding Remarks

The techniques presented within this chapter have provided an initial examination of the visualisation techniques that may be used to extract knowledge from the data within the FCS. Visualisation has been shown to be a powerful tool since it supports the exploration of the FCS and increases the domain knowledge of the designer. In this sense visualisation sits comfortably within the iterative foundations of COGA. However, a drawback of visualisation is overplotting. It is suggested that the use of a dynamic or virtual

---

\(^1\) Non-immersive VR includes the use of mouse controlled navigation through a multi-dimensional environment, presented upon a display medium such as a monitor [Beier 1999].
environment where the visualisation space may be expanded, distorted and its orientation
manipulated would significantly reduce if not eliminate such drawbacks.

It should be noted that no effort has been made to investigate the use of datamining tools
such as decision trees, clustering methods, neural networks and so on. Nevertheless as
both the dimensionality of the design domain and the number of extracted solutions
increases, the use such techniques become increasingly necessary. The more interested
reader may should refer to [Adrianns et. al. 1996] for a comprehensive review of current
data mining techniques. However caution should be exercised since the use of such
techniques largely removes the designer from any of the decision processes, and may
subsequently reduce the confidence placed upon derived relationships.


CHAPTER 9

Conclusions

9.1 Conclusions

This thesis has explored the support of conceptual engineering design by the use of Cluster Oriented Genetic Algorithms (COGAs). This is achieved by decomposing the design domain into regions of high performance, which indicate to the designer optimal or novel design directions. Extracted information such as constraint violation, multi-objective criteria etc. may be used in conjunction with design team knowledge to assess the utility of each identified region. COGAs are therefore used within highly interactive and dynamic environments, where design variable ranges, objective weightings and design objectives may be changed to promote an iterative process that increases overall design domain knowledge. The application of COGA to four commonly encountered design scenarios demonstrates the power of the approach, whilst its integration with a Graphical User Interface and the Interactive Evolutionary Design Station gives examples of possible operating environments.

Although providing an initial solution to promoting diversity and increased search space sampling, the effectiveness of variable mutation is questioned. A series of “off the shelf” and novel techniques have been developed to investigate the utility of replacing variable mutation. A set of performance measures is proposed that assess every desired characteristic of a regional decomposition algorithm. The test domain is provided by three functions where regions of high-performance are defined apriori. Two functions contain two dimensions whilst the third is developed to overcome the limitation of testing regional decomposition algorithms within one or two dimensions. This innovative approach defines
high performance regions within sub-planes, which are then summated to produce an overall function consisting of any number of global high performance regions.

The replacement of variable mutation (vmCOGA) with an algorithm using a variation of uniform crossover and incest prevention shows little improvement. However replacement with Deterministic Crowding and also the ECOGA indicates some utility under 2d-test conditions. Nevertheless, when applied to the 6d-test function, the performance of both algorithms diminishes drastically. ECOGA search converges rapidly upon a limited number of regions, sacrificing all global exploration. Whilst, DC search exhibits lower levels of convergence and reduced local exploitation of high performance regions. Such failures may be tempered to a degree by the hybridisation of both algorithms, but the need to reduce or increase population size when moving from DC to ECOGA search and vice versa, introduces considerable problems and does not produce any improvement over vmCOGA under higher dimensionality conditions. Although an advantage of the DC and ECOGA algorithms lies in the fact that they do not contain parameters that require tuning before search. It is suggested that this introduces considerable limitations since it removes the ability to control convergence during search.

The injection of Halton individuals (HiCOGA) and the use of spatial selection (SsCOGA) may be used to control the degree of search space sampling and the convergence evident within the FCS. As the number of injections or spatial selections increase, FCS convergence decreases and search space sampling and the subsequent set cover within regions of higher sensitivity increases. Of the two spatial selection density metrics, there is little empirical evidence to indicate the superiority of either. However the nearest neighbour Euclidean distance metric does not require the setting of any parameters and is therefore preferred. The effects of Halton injection appear to be more pronounced when compared with spatial selection and reduce FCS convergence to a greater degree, hence
similar Halton injection and spatial selection vectors do not produce comparable results. A further effect of both techniques is an increase in GDE and set cover standard deviation indicating increased algorithm sensitivity.

The control mechanism eliminates the need to set \( H_i \) and \( s_s \) and also allows for a more accurate comparison between both techniques and \( \text{vmCOGA} \). Two further algorithms are presented: \( \text{amCOGA} \) adapts the mutation rate during search and is therefore the closest relative to \( \text{vmCOGA} \); \( \text{mssCOGA} \) adapts both spatial selection and mutation and thus eliminates the need to set a baseline mutation rate as is the case with \( \text{SsCOGA} \). The population convergence metric is an excellent feedback measure since its profile in every simulated search scenario changes little as dimensionality increases. However, a drawback is the need to specify the number of generations over which to calculate the convergence rate. Although the control mechanism is very rudimentary and little attention is given to its transient or steady state response, it performed well for the specified parameters, maximum control error, maximum control change etc. The subjectivity of setting the target convergence rate is reduced by basing upon multiples of the convergence rate measure with \( \text{vmCOGA} \). Empirical studies upon the 6d-test function indicate comparable levels of convergence within the FCS of each adaptive algorithm for any given target convergence rate. Further results support the conclusion that Halton injection and spatial selection increase the sampling of the search space. At target convergence rates of 1.5 that of \( \text{vmCOGA} \), mutation alone is unable to support the exploration of the more sensitive and isolated regions within the search space and search converges upon a limited number of more robust areas. The use of Halton injection and more notably spatial selection maintains such sampling and subsequently produces higher GDE without appreciable loss in either set cover or sampling within sensitive regions. \( \text{SsCOGA} \) is found to outperform \( \text{vmCOGA} \) in upto 94% of all high performance regions.
Although the general appearance of convergence metric profile changes little with increasing dimensionality, there is a gradual decrease in convergence rate (due to an increase in the number of generations per search stage). Consequently, a series of calibration curves are proposed that determine the equivalent vmCOGA convergence when applied to functions of increasing dimensionality. When applied to the two dimensional test functions using a calibrated target convergence rate equal to that of vmCOGA, increased exploration of the more sensitive regions is again noted in algorithms using spatial selection and Halton injection. However reduced set cover results in lower numbers of solutions in all regions when compared with vmCOGA. It is therefore recommended that vmCOGA or amCOGA (using a target convergence rate equal to vmCOGA) should be used in the unlikely scenario where 2d models are investigated in the real-world domain. When the adaptive algorithms are applied to miniCAPS, TBCOM and Keane’s function, reduced set cover is not evident, suggesting that the reduced performance when applied to 2d functions will not be present when applied to test function of increased dimensionality. However caution must be exercised since the lack of explicit knowledge relating to the nature and location of high performance regions in such functions means that the increased sampling caused by Halton injection and spatial selection can not be corroborated.

The Dynamic Adaptive Filter (DAF) increases algorithm robustness by reducing both the set cover and GDE standard deviation. The five library PDFs are chosen to cover the whole envelope of projected population fitness distributions, the use of both Kolmogorov-Smirnov and $\chi^2$ metrics allow for application to any population size. Integration with vmCOGA, SsCOGA and mssCOGA indicate that the DAF significantly increases algorithm robustness without any discernible reduction in the sampling of the search space. A side effect of the DAF is a slight increase in the severity of the filtering process characterised by higher GDE and lower set cover. This may be alleviated by a minor
reduction in the filtering vector. Such effects are therefore trivial and outweighed by the significant increase in algorithm robustness.

The final section of the thesis explores methodologies for presenting the extracted data within the FCS to maximise knowledge discovery. In the real-world environment the importance of this can not be overstated, since it provides the primary link between the design team and the design model. The practical utility of COGA therefore depends heavily upon the efficiency of this interface. It is suggested that the simplicity of visualisation coupled with the fact that the designer forms the central decision making element make this approach the most viable means of knowledge discovery within COGA at this time. It is further suggested that the use of interactive elements such as linked brushing and the rotation, translation and distortion of the visualisation space greatly add to the ease, speed and quality of knowledge gain. Consequently, chapter eight may be seen as an analysis of the visualisation techniques that should be included within packages using COGA such as the IEDS.

9.2 Future Research Directions

There are many areas identified within this thesis that warrant further investigation. The assessment of the adaptive COGA algorithms and the identification of optimal target convergence rates when applied to functions of higher dimensionality should be considered a priority. The 6d-test function has provided an initial means to support such investigations, however a geometric increase in the number of global high performance regions with increasing dimensionality makes the approach unfeasible for higher dimensionality search domains\(^1\). A more efficient solution may lie in the use of a mapping

\(^1\) A 10d function consisting of five sub-planes each with three planar high performance regions will contain 243 global high performance regions.
algorithm, such as principal component analysis to linearly combine dimensions thereby reducing the function to lower dimensionality. A comparison of the high performance regions within the 6d-test function before and after mapping may provide some indication of the suitability of such methods.

During the development of the control method little attention was paid to the transient or steady state response of control, this should be rectified. Although the suggested values for the number of interpolated points, maximum error, maximum control input etc. appear to produce acceptable control, the use of more advanced techniques which do not require such parameters (such a fuzzy control) should be regarded as a necessity.

To date the adaptive algorithms have all been used in a generational sense where the search parameters are only adapted at the end of each generation. It is suggested that the use of a steady state configuration may increase the performance of control, and therefore warrants further investigation. The elimination of the calibration curve and the need to set the target convergence rate should also be regarded as a long-term goal.

A significant area of future work lies with knowledge discovery and the GUI / COGA framework. The use of linked brushing, rotation, distortion etc. have been shown to be a necessity for the efficient exploration of the visualisation space and should therefore be more rigorously investigated, the role non-immersive Virtual Reality may play in such an environment should also be addressed. It is acknowledged that as the as the size of the FCS and the dimensionality of the design domain increase the use of datamining tools such as decision trees, clustering methods may become necessary and warrant further investigation. Finally, the design of the human / computer interface used within the GUI should also be seen as an important area of future research, since this greatly effects the performance and user friendliness of COGA within the real-world environment.
Appendix A: Test Function 2

\[ 0.0 \leq x \leq 0.8 \]
\[ 0.0 \leq y \leq 0.8 \]

\[ a_1 = -10(x - 0.1)(x - 0.6)^3 \]
\[ b_1 = (x - 0.6)^2 + (y - 0.3)^2 \]
\[ f_1 = a_1 + \frac{0.04}{b_1 + 0.1} \]

\[ a_2 = (x - 0.4)^4 + (-0.2)^4 \]
\[ f_2 = \frac{0.35}{a_2 + 1} \]

\[ a_3 = (x - 0.2)^6 + (y - 0.2)^4 \]
\[ f_3 = \frac{0.1}{a_3 + 0.1} \]

\[ a_4 = (x - 0.77)^2 + (y - 0.77)^2 \]
\[ f_4 = \frac{0.022}{a_4 + 0.01} \]

\[ f_5 = f_1 + (f_2 \times f_3 \times f_4) \]

\[ a_6 = (x - 0.1)^2 + (y - 0.7)^3 \]
\[ f_6 = f_5 + \frac{0.025}{a_6 + 0.1} \]

\[ f_7 = f_6 + 0.3e^{-20x} \sin(40x + 5) \sin(50y) \cos(80xy + 4) \]

\[ z = \frac{|f_1|}{-0.01606e^{-1} + 0.65205} + \frac{\sin(15x + 3) \sin(10y)}{2} \quad \text{for } z \geq 0.0 \]
Appendix B: 6d Test Function – Plane 1

\[ 0.0 \leq a \leq 1.0 \]
\[ 0.0 \leq b \leq 1.0 \]

\[ x_{11} = \frac{0.41}{0.04 + \frac{(0.7 - b)^2}{0.04} + 1} \]

\[ x_{12} = \frac{0.42}{0.0225 + \frac{(0.75 - b)^2}{0.09} + 1} \]

\[ x_{13} = \frac{0.1936}{0.2025 + \frac{(0.06 - b)^2}{0.09} + 1} \]

\[ z_1 = x_{11} + x_{12} + x_{13} \]
Appendix C: 6d Test Function – Plane 2

0.0 \leq c \leq 1.0
0.0 \leq d \leq 1.0

IF(c<0.5)AND(d\geq0.5)

\{ 
\begin{align*} 
x_{21} &= \frac{0.5}{(0.25-c)^2 + (0.75-d)^2} \\
x_{22} &= 0.5 - e \frac{\cos(48\pi) + \cos(48\pi)}{14.8} \\
z_3 &= z_2 - x_{21} 
\end{align*} 
\}

IF(c\geq0.5)AND(d\geq0.5)

\{ 
\begin{align*} 
z_2 &= \frac{0.35}{(0.75-c)^2 + (0.75-d)^2} \\
\end{align*} 
\}

IF(c<0.5)AND(d<0.5)

\{ 
\begin{align*} 
z_2 &= \frac{0.35}{(0.25-c)^2 + (0.25-d)^2} \\
\end{align*} 
\}

IF(c\geq0.5)AND(d<0.5)

\{ 
\begin{align*} 
x_{21} &= \frac{0.5}{(0.75-c)^2 + (0.25-d)^2} \\
x_{22} &= 0.5 - e \frac{\cos(24\pi) + \cos(24\pi)}{14.8} \\
z_3 &= x_{22} - x_{21} 
\end{align*} 
\}
Appendix D: 6d Test Function – Plane 3

0.0 ≤ e ≤ 1.0
0.0 ≤ f ≤ 1.0

IF(e<0.6) AND (f≥0.5)
{
    z₃ = 1.8e² + 3f²
}

IF(e≥0.6)
{
    IF(f<0.8)
    {
        z₃ = 0.635f
    }
    IF(0.8 ≤ f<0.85)
    {
        z₃ = 0.5
    }
    IF(f≥0.85)
    {
        z₃ = \frac{10f + 10}{3}
    }
}

IF(e<0.6) AND (f≥0.5)
{
    z₃ = 3.75(0.6 - e)(f - 0.5)
}

IF(z₃>0.5)
{
    z₃ = 0.5
}
Appendix E: Keane’s 20d Bump Function

\[ 0 < x_i < 10 \quad i = 1, \ldots, n \]

\[
    f(x_i) = \frac{\sum_{i=1}^{n} \cos^4(x_i) - 2\prod_{i=1}^{n} \cos^2(x_i)}{\sqrt{\sum_{i=1}^{n} i x_i^2}}
\]

subject to,

\[
    \prod_{i=1}^{n} x_i > 0.75 \quad \text{and} \quad \sum_{i=1}^{n} x_i < \frac{15n}{2}
\]
Appendix F: Hastings’s best approximation to the standardised normal distribution [Hastings 1955]

\[ p(X \leq x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} = 1 - \Psi(C_1 t + C_2 t^2 + C_3 t^3 + C_4 t^4 + C_5 t^5) \]

where,

\[ \Psi = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} \]

\[ t = \frac{1}{1 + P_X} \]

\[
\begin{align*}
P &= 0.2316419 \\
C_1 &= 0.31938153 \\
C_2 &= -0.356563782 \\
C_3 &= 1.78147937 \\
C_4 &= -1.821255978 \\
C_5 &= 1.330274427
\end{align*}
\]
Appendix G: Approximating $\alpha$ and $\theta$ for the Weibull PDF

The proof of this approach is rather involved and can be best illustrated by example, however the interested reader may refer to [Scheaffer et. al. 1995].

Consider a population of 5 individual observations of some variable $x$ ($n=5$) sorted in ascending order, $\alpha$ and $\theta$ are calculated as follows,

1. Construct the following table

<table>
<thead>
<tr>
<th>Rank $(i)$</th>
<th>Observation $(x)$</th>
<th>ln(x)</th>
<th>$\frac{i}{n+1}$</th>
<th>ln ln $\left[ \frac{1}{1 - \frac{i}{n+1}} \right]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>22</td>
<td>3.091</td>
<td>0.167</td>
<td>-1.699</td>
</tr>
<tr>
<td>2</td>
<td>24</td>
<td>3.178</td>
<td>0.333</td>
<td>-0.903</td>
</tr>
<tr>
<td>3</td>
<td>24</td>
<td>3.178</td>
<td>0.429</td>
<td>-0.579</td>
</tr>
<tr>
<td>4</td>
<td>28</td>
<td>3.333</td>
<td>0.667</td>
<td>0.095</td>
</tr>
<tr>
<td>5</td>
<td>30</td>
<td>3.401</td>
<td>0.833</td>
<td>0.582</td>
</tr>
</tbody>
</table>

2. Draw a graph, plot ln(x) upon the x axis and ln ln $\left[ \frac{1}{1 - \frac{i}{n+1}} \right]$ upon the y axis.

3. Deduce by regression or other means the line of best fit through the points.

4. The gradient of the line is $\alpha$.

5. $\theta$ can be found from,

$$\ln(\theta) = \text{intercept}$$
References


