EVOLUTIONARY ALGORITHMS AND AGRICULTURAL SYSTEMS

David G. Mayer

SPRINGER SCIENCE+BUSINESS MEDIA, LLC

EVOLUTIONARY ALGORITHMS AND AGRICULTURAL SYSTEMS

THE KLUWER INTERNATIONAL SERIES
IN ENGINEERING AND COMPUTER SCIENCE

EVOLUTIONARY ALGORITHMS AND AGRICULTURAL SYSTEMS

by

David G. Mayer
Queensland Beef Industry Institute, Australia

SPRINGER SCIENCE+BUSINESS MEDIA, LLC

ISBN 978-1-4613-5693-6 ISBN 978-1-4615-1717-7 (eBook)
DOI 10.1007/978-1-4615-1717-7

Library of Congress Cataloging-in-Publication Data

A C.I.P. Catalogue record for this book is available
from the Library of Congress.

Originally published by Kluwer Academic Publishers in 2002
Softcover reprint of the hardcover 1st edition 2002

Printed on acid-free paper.

Contents

Preface

Systems research is increasingly being used to investigate and analyse a wide range of real-world problems, including agricultural production systems. Given a valid, verified model of a particular system, optimisation is a logical complement to the modelling exercise. Usually, this takes the form of maximisation of some measure of the system's performance, such as total production or economic gross margin. This book deals with the practical application of optimisation techniques, particularly evolutionary algorithms, to the study and management of these agricultural systems. It should prove useful to practitioners applying these methods to the optimisation of agricultural or natural systems, and would also be suited as a text for systems management, applied modelling, or operations research university subjects. Basic knowledge in systems research, along with some computing and programming skills, are assumed.

Models of agricultural systems range widely on both temporal and spatial scales. Farm-level systems have typically been investigated, but models also range out to regional, industry and national scales. Short-term (within-year) profitability and cash-flow issues are common, but the time-frame can be extended to a hundred years or more, to investigate sustainability and long-term effects. In addition to the 'direct' economic maximisation of agricultural systems, optimisation methods have also seen use in the calibration of internal model parameters to observed data, maximising the rate of genetic gain in livestock, in agricultural allocation and scheduling problems, and in the analysis of sustainability issues in natural systems management.

Agricultural models present a number of difficulties with regard to optimisation. These problems include complex relationships which are not conducive to the simpler forms of economic modelling (such as linear programming); biological variability, which usually requires a stochastic

model; the identification of suitable variables to optimise; the high degree of complexity in these systems, which translates to high dimensionality of the search-space; frequent interactions between the effects of the various (assumedly independent) management options; cliffs and discontinuities in the search-space (where the system is over-utilised, and 'crashes' both biologically and economically); and the presence of multiple local optima, caused by very different combinations of management options having similar economic outcomes.

Any selected optimisation method is required to deal with all these problems, and reliably return the solution for the global optimum (or a value suitably close to this). Generally, evolutionary algorithms (including genetic algorithms, evolution strategies, and hybrid methods) have proven superior for this task. Depending on the algorithm and the type and usage of the model, some problems do remain, however evolutionary algorithms contain a number of advantageous features which largely circumvent these. For the alternate optimisation techniques (including gradient and direct-search methods, simulated annealing, and the tabu search strategy), these difficulties often prove insurmountable. Published studies on the application of all these methods to agricultural systems are contrasted and compared, in terms of quality of the final solution and rates of convergence.

Finally, the listed applications are drawn together into an overview, and the more successful genetic algorithm methodologies and parameters are discussed. This identifies combinations which are likely to provide robust performance, given any similar future system being investigated. Directions of profitable future research are also outlined.

Acknowledgments

This research was conducted whilst I was an employee of the Queensland Department of Primary Industries, and I gratefully acknowledge the support of this organisation, along with the encouragement of my supervisors, Drs Pat Pepper, Peter O'Rourke and Tony Swain.

Sincere thanks are also due to quite a number of scientists for assistance, namely

- Professors John Belward and Kevin Burrage of the Centre for Industrial and Applied Mathematics and Parallel Computing, University of Queensland, for their guidance and expertise with optimisation techniques.
- Personnel in the Climate Impacts and Natural Resource Systems section, within the Queensland Department of Natural Resources and Mines. In particular, Greg McKeon, Grant Stone, Wayne Hall, Samantha Watson and Neil Flood all provided valuable assistance with the development of these agricultural models.
- The practitioners who have made their optimisation algorithms available to the general community on the world-wide web, namely John Grefenstette (GENESIS), Lester Ingber (ASA) and Henrik Widell (GENIAL).
- Colleagues in the Modelling and Simulation Society of Australia and New Zealand who assisted with discussions and critiques, in particular David White, Mark Howden, Chris Dake and Robert Sherlock.

Finally, I give special thanks to my wife Ailsa, and children Nicole, Kerryn, Janice, Gregory and Cristie for their support and understanding.

Chapter 1

RATIONALE FOR SYSTEMS MODELLING

This introductory chapter outlines the documented benefits of the systems research approach, and the steps and methodologies typically used here. Potential discrepancies between the modelled and real-world systems are discussed, along with interpretational issues. The various types of models used for the study and optimisation of agricultural systems are outlined. These include the widely-used and much published linear and mathematical programming methods, which however do tend to be constrained representations of the real world, and can give poor results. As an alternative approach, general (unstructured) simulation models are promoted as a more flexible and realistic method of representing the system being modelled.

1. INTRODUCTION

With the advent of more powerful computers, the science or art of simulation modelling has become more commonplace. Simulation of a system is the construction and operation of a model which is a valid representation of the system. Physical models have long been used to investigate a variety of problems. For example, the effectiveness of design in boats and aircraft has been tested by the behaviour of smaller-scale models in tanks or wind-tunnels. Also, erosion and sedimentation models of proposed harbour and canal developments are commonly used, with some of these physical models covering hundreds of square metres. Despite the cost involved, it is obviously advantageous to gain an understanding of potential problems and solutions prior to spending millions of dollars on such a development.

Computer simulation models of agricultural systems are essentially the same. They are a representation or abstract of reality, expressed in mathematical and logical terms. As such, they can never fully represent reality, and must be imperfect. The only perfect model is reality itself. For example, the best pasture production model, taking account of rainfall, runoff, through-drainage, water use efficiency, humidity, solar radiation, soil fertility, mobile nutrients, inter- and intra-species competition, etc. will still not produce accurate results if excessive rainfall produces a flood which submerges the pasture for a period of time (unless the modeller has allowed for this occurrence). Despite these shortcomings, it is the intention of most modellers to construct a model which will simulate reality as well as is possible, and which may be assumed valid under most circumstances. In particular, the model should work well in situations similar to those where the modelled results will be extended to and used in the real-world system.

The logistics and mechanics of model construction are complex, covering problem definition; construction of systems diagrams; selection and estimation of key parameters, pathways and relationships; data availability; programming approach; hardware and software requirements; and verification and validation. Figure 1 represents a typical systems diagram (from our series of simulation studies, Mayer 2000) of an agricultural system, from which a systems model can, and has, been built. These systems research methodologies have been well described in a range of introductory and advanced texts (Dent and Blackie 1979, Law and Kelton 1982, Bratley et al. 1987, Kleijnen 1987, Ripley 1987). It is not the intention here to review this diverse field, but rather to consider applications of agricultural models from a practical point of view.

The development and proving of a systems model can be lengthy and expensive, with no guarantee of success, so a critical analysis of suggested applications should be undertaken prior to its commencement (Bennett and MacPherson 1985). In systems where a valid model can beneficially be constructed, a range of advantageous uses exists, including -

1. Models can be used for manipulations and experiments which would be impractical, too expensive, too lengthy, or impossible in the real world. When a verified, valid model has been constructed, a wide range of experiments can be conducted at very minimal cost compared to traditional agricultural research.
2. In the real system, complex interactions often exist. Multiple runs of a model can be used to identify and quantify these, further justifying this approach as an alternative to reductionist research which may only consider one dimension of the overall problem.

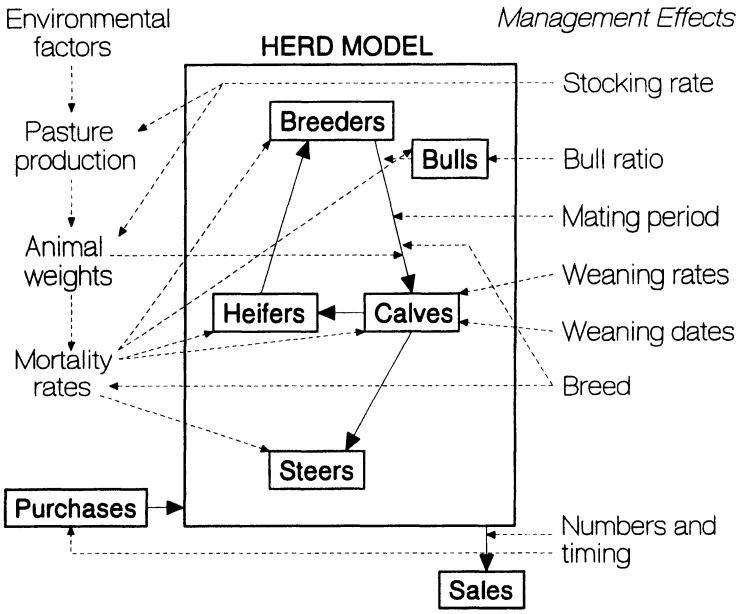

Figure 1. Systems schematic of a beef property model for northern Australia.

3. The best management strategies can be identified rapidly, via optimisation. This, and probably also a number of predicted near-optimal strategies, can then be field-tested.
4. The long-term effects of options can be evaluated. In some cases, significant results may only appear after a number of years, and a traditional-style experiment may be written off prior to this.
5. In modelling, the researcher has control over environmental as well as experimental conditions, and this can be useful in determining long-term strategies. Pre-determined meteorological conditions can be tested, as opposed to the conditions which naturally and variably occur each year in agricultural experiments.
6. Hypothetical and exploratory situations such as climate change can be investigated, to prepare for such eventualities. The overall stability of the system under such conditions can be researched.
7. An insight is obtained into the relative importance of the variables and factors of the system. Unimportant variables can subsequently be

ignored, and the key variables and relationships can be more accurately estimated from research, if required.

One implicit assumption of the modelling approach is that any results obtained via the model will also apply in the real-world system, i.e., the 'best' way to manage the modelled agricultural system will also be the best way to run this system. This does not always occur – for example, with the modelled autonomous orchard sprayer, Cho and Lee (2000) simulated a 68% improvement over the baseline scenario. However, this did not translate across to the real world, due to engineering problems including tyre slip and hydraulics response time. If these were solved, we would then expect the simulated result to apply. Conversely, Davies et al. (2000) identified an optimal combination of additives for silage preparation which outperformed the accepted industry standards, both in the model and then when applied to the real world. As any model is an abstraction of reality, perfect agreement with the modelled system will never be obtained. However, if the model is verified and validated for the purposes of the study, then the degree of difference should be small, and managers can confidently take modelled results across into the real world.

'Agricultural Systems' and other applied journals have published numerous examples of the use of models, both explicit (the study is defined as a simulation, with the usual terminology) and implicit (where logical or knowledge-based descriptions of the studied system form an analytical model for solution). In many of these examples, the model is used to analyse and predict the likely effects of managerial changes, to improve or optimise some measure of production, profitability or overall utility.

2. TYPES OF SYSTEMS MODELS

In 1826 von Thünen outlined a simplified model of an agricultural production area (Stevens 1968). Since this time agricultural economists have modelled and optimised numerous systems (Judge et al. 1977). In agriculture alone, 'the literature (of optimisation theory and methods) is indeed by now so vast as to preclude a comprehensive survey' (Day 1977). This statement was made over twenty years ago, when Day and Sparling (1977) tabulated around 350 individual references, and many more have appeared since. The vast majority of these types of studies use the broad field of mathematical programming, including linear programming, integer programming, quadratic programming, nonlinear programming, and dynamic programming.

MATHEMATICAL PROGRAMMING METHODS

In these studies, the modelled system is represented by a framework of mathematical equations. Whilst theoretically elegant and conducive to solution, the application of these methods to large, complex real-world systems has definite problems. Mathematical programming methods generally rely on smooth (or even linear) functions to adequately describe the system. They have difficulties dealing with extreme behaviour and interactions between variables (Gabbert et al. 1991, Michalewicz and Fogel 2000), as well as coping with dynamic interactions between the system's key variables over time (Hayes et al. 1997). Linear programming models can be iteratively re-formulated and re-run to accommodate nonlinearities and dynamic problems (King and Logan 1964), but this is obviously inefficient. Most agricultural systems are spatially heterogenous, yet these methods assume homogeneity within defined regions (MacKinnon 1976). The almost infinite number of regions required to satisfy this cannot be handled computationally, so simplifying assumptions often need to be made (Lazarus and Dixon 1984, Honghai 1987) or special features incorporated to make the problem tractable (Pratt et al. 1986). Also, the smooth nature of nonlinear programming makes them prone to finding local rather than global optima (Kuo et al. 1992, Messine et al. 1996). In practical applications with simulation models, mathematical programming methods are realistically limited to problems of only moderate size and complexity (Michalewicz 1996). In evaluating rangeland investment decisions via nonlinear programming, Lambert and Harris (1990) assumed homogeneity across a ranch of 78 000 acres, and still required a supercomputer to obtain a solution. On a dynamic crop irrigation problem using linear programming, de Juan et al. (1999) experienced difficulties handling the complexities of multiple farms and paddocks.

Regarding agricultural systems models, increasingly more researchers are finding that mathematical programming methods are not well suited. A summary of these studies, along with the optimisation method that was successfully adopted, is listed in Table 1. Only one of these studies gives a direct comparison of optimisation results – in the English Channel fisheries, Mardle and Pascoe (2000) reported that their genetic algorithm found an optimal combination which was 4.3% higher than the linear programming solution.

Overall, the usefulness of mathematical programming techniques appears limited by the restrictiveness of their formulation, which is required to facilitate computational solution (Gill et al. 1981). In some situations these methods contribute adequate strategies, but 'there are many cases where these assumptions lead to very poor (high cost) solutions to the real problem' (Gabbert et al. 1991, p. 430).

Table 1. Agricultural systems studies where mathematical programming methods could not be successfully utilised.

Reference	Agricultural system	Unsuccessful maths. programming method	Optimisation method adopted
Annevelink (1992)	Greenhouse allocation	Linear programming	Genetic algorithm
Bos (1993)	Forestry zoning	Linear programming	Simulated annealing
Lockwood and Moore (1993)	Harvest scheduling in forestry	Linear and mixed integer programming	Simulated annealing
Botes et al. (1996)	Crop irrigation problem	Dynamic programming	Simplex
Hayes et al. (1997)	Mate selection in breeding program	Linear programming	Genetic algorithm
Parsons (1998)	Silage harvesting scheduling	Dynamic programming	Genetic algorithm
Kuo et al. (2000)	Crop irrigation scheduling	Linear programming	Genetic algorithm
Lu and Eriksson (2000)	Harvest scheduling of forestry stands	Linear and mixed integer programming	Genetic algorithm
Mardle and Pascoe (2000)	Fisheries management	Linear programming	Genetic algorithm
Moore et al. (2000)	Silviculture harvest scheduling	Dynamic programming	Genetic algorithm

GENERAL METHODS

The alternative to mathematical programming is to first develop 'general' simulation models, which have no set restrictions or requirements. These models can then be optimised, using any of the range of available algorithms. The models are still a mathematical and logical representation of the system under study, and vary widely in scope, scale and complexity (Thornton and McGregor 1988). This open-minded nature enhances their general usefulness and applicability, as models can continue to be refined and improved until satisfactory. For example, if a pasture agronomist investigating a beef production model decided that the simple empirical formula describing legume dynamics was inadequate, it could be replaced by a detailed inter-plant competition model. The refinement of this module may take some extra tuning and computation time, but the rest of the model and all output, graphical and optimisation routines would remain unchanged. The entire model would not have to be reformulated, as would be required under a mathematical

programming approach (if it could actually be used in this situation). Allen and McGlade (1986) show how an initially simple fisheries model can be expanded to allow more realistic and important real-world effects to be modelled, first incorporating the interactive responses of the system's exploiters (the fishing fleet), and then spatial dynamics (Allen and McGlade 1987).

Having invested considerable time and resources in the planning, formulation, verification and validation of a 'general' model which is deemed to adequately simulate the target system, the modelling team is faced with the next problem of optimisation. Large, multi-dimensional problems require targeted and efficient optimisation routines. Searching the feasible space of available management options and coming up with the global (as opposed to a local) optimum is a difficult task, especially so as the dimensions and complexities of the problem increase (Meadows and Robinson 1985). Evolutionary algorithms appear well-suited for this task, as they are amongst the most efficient of the available optimisation methods. Also, by carrying a population of solutions, they allow the identification and investigation of near-optimal strategies (Parsons 1998), which may also be of interest to the system's manager.

Chapter 2

AGRICULTURAL SYSTEMS MODELS

The types of simulation models used with agricultural systems vary widely in terms of scale, scope and purpose. They range from the micro-level (animal genetics and physiology), through paddock and farm-level models, to large regional or national systems. These various applications are illustrated, using published examples from this field. Next, the proven advantages and uses of systems research in agriculture are outlined. The choice of which variable to optimise depends on the purpose of the study, and include different measures of deviance (for model tuning applications), the gross production or margin (profitability) of the modelled system, or some utility function further incorporating risk or other important factors. This leads on to the consideration of multi-objective optimisations, where the multiple competing outcomes are frequently negatively correlated. By considering the types of results that end-users expect, the optimisation requirements of this approach are considered. Finally, the particular types of problems which agricultural systems models pose are listed and discussed. Practical methods of dealing with these problems are outlined, again using agricultural examples from the literature.

1. FEATURES AND METHODOLOGIES

Many general models of real-world systems concentrate on discrete, controllable problems, such as transport systems, service queues, aircraft design, factory layout and scheduling, and engineering designs. On the other hand, models of agricultural enterprises range widely in scope and purpose (Mayer 1985). These go from sub-farm and whole-farm simulations through to national or international industry systems, as exemplified in many issues

of the journal 'Agricultural Systems'. Fortunately for systems researchers working in this field, the modelling approach is applicable across the whole range of scales (Thornton and McGregor 1988).

At the lower end of the spectrum, models have been used at the micro-level, for example genetic improvement studies (Hayes et al. 1997, Meszaros et al. 1999) and within-animal physiology and nutrition (Freer et al. 1997). Models can be used to investigate pasture dynamics or cropping systems (Hammer and White 1992), on a within-paddock basis. The next level up is the one most frequently studied and published in the literature (as will be listed in the following chapters), namely models of a single farm or agricultural enterprise. Typically, these farms are owned and managed by a single person or family, and form a discrete economic enterprise which can be studied and optimised. These systems occur world-wide, and range across small-plot mixed farming, intensive livestock (for example, piggeries, poultry, dairying, cattle feedlots, and egg production), plantations, grain and field crops, and large-scale grazing or mixed enterprises. Single properties can also be integrated into multi-farm systems. For example, in Australia there are several large pastoral companies which own a number of beef producing properties, often in different geographical and environmental regions to take advantage of climatic and vegetation variability. The management, nutrition and flow of animals from birth through to marketing forms an integrated operation. Silviculture applications typically focus on whole-forest or area management, investigating the optimal harvesting schedules over time-frames which usually cover a number of decades (Roise 1990, Lockwood and Moore 1993, Lu and Eriksson 2000, Moore et al. 2000).

Regional or whole-system studies are also common in agriculture, and include models of the grain distribution system of the USA (Koo and Thompson 1982), the dairy industry of north-eastern USA (Pratt et al. 1986), animal production in the savannas of Colombia (Thornton 1988), the wheat grazing systems of the USA southern plains (Rodriguez et al. 1990), the northern Australian grazing industry (White et al. 1998), and fisheries in Western Australia (Watson and Sumner 1999) and the English Channel (Mardle and Pascoe 2000). Agricultural models covering whole nations (or even the global level) remain possible, but would tend to be difficult to implement properly because of the diverse and heterogeneous nature of any targeted system on this scale.

Systems models offer advantages in the study of many scientific and commercial areas, including agriculture (Rickert and Winter 1980). Given the complexity of these systems, a modelling approach is the only practical method to evaluate the multitude of dynamic interactions (Hammer and White 1992). In this context, models can be used both strategically (where the best long-term

overall strategy for the system is to be determined), and tactically (shorter-term, where the best tactics for the current situation can be evaluated). This tactical focus has proven valuable in the extension role, for example training producers to think about their farms and evaluate the likely responses given different 'what-if' scenarios (Gillard and Monypenny 1988, Rossing et al. 1997). For a number of agricultural systems, the results of model investigations have been adopted by primary producers (Hammer and White 1992). This is particularly so when the model results confirm the current industry best practice, as in the New Zealand sheep industry (Barioni et al. 1999a). Conversely, Parker et al. (1999) point out that the uptake of decision support models and results by New Zealand farmers has been slower than expected. As an extension to the evaluation of the immediate (tactical) effect of interacting management options on the farming enterprise, longer-term strategies and dynamic interactions over any reasonable time-horizon can also be investigated (Foran et al. 1990, Mayer et al. 1998a, Howden et al. 1999).

The use of models for extrapolative purposes can include the investigation of strategies which are have yet to be used in practice, such as determining policies to minimise agricultural greenhouse emissions (Kulshreshtha et al. 2000) and environmental impact (Pluimers et al. 2000), or situations which are yet to occur, including climatic and environmental change (White et al. 1996, Hall et al. 1998, Howden et al. 1999). Models can also be used to recreate and estimate historical events which were not explicitly recorded, such as major land degradation events (McKeon et al. 1990, Stafford-Smith and McKeon 1998). In some of these cases, modelling studies can have a major impact on society, in that they have been used to determine national agricultural policy (White et al. 1998).

One remaining concern is the applicability of these models to their respective real world populations, and the potential problem of extrapolation. Models are typically developed using data at hand, which is assumed as being applicable to the targeted population. This is not always the case – Figure 1 shows a comparison between modelled mortality rates (which were based on data from research stations) versus rates observed in on-farm trials across the state of Queensland, from our studies (Mayer et al. 1999c). The estimated mortality rates here are taken as the independent axis, to conform with statistical assumptions (Mayer and Butler 1993, Mayer et al. 1994a). This figure indicates a high degree of bias. Similarly, genotype by environment interactions are common in agricultural systems (Richardson and Hahn 1994), hence common parameters cannot be used across all systems. Production models need to be adapted and tuned to each different vegetation community or environment before being used, as done in Rickert and Winter (1980) and Clark et al. (2000).

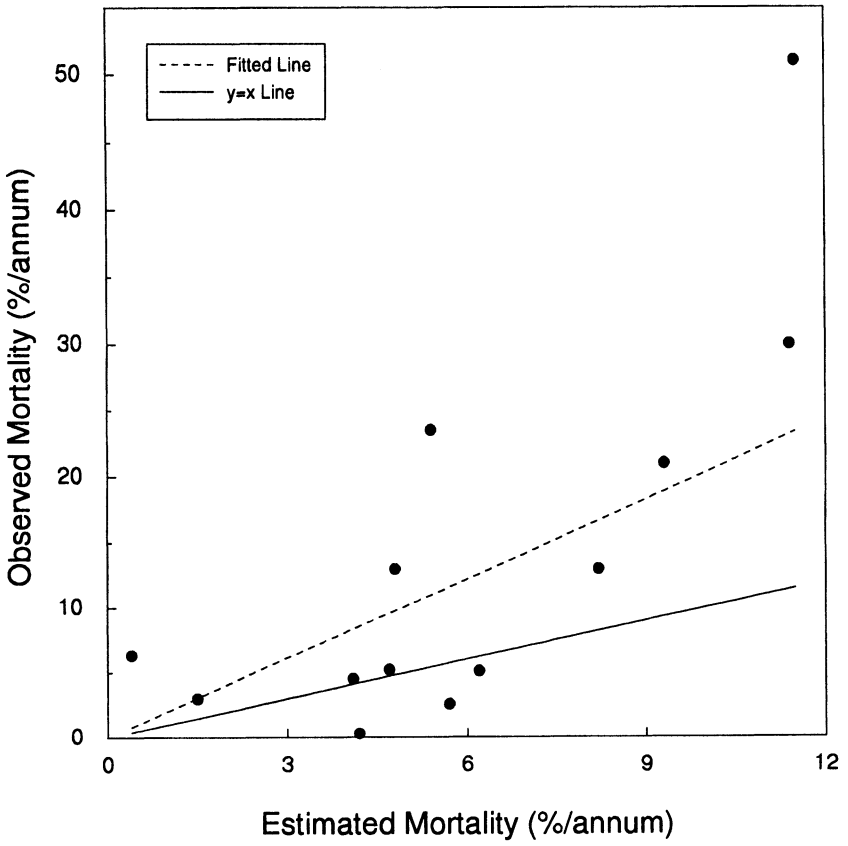

Figure 1. Illustration of bias – the model was based on research stations data, whereas the actual mortalities were observed on farmers' properties.

2. OPTIMISATION REQUIREMENTS

There are a number of different end-users of agricultural models, with each having specific requirements. Researchers typically wish to investigate the system thoroughly, to make sure it is a realistic representation of the real world, and arrive at valid conclusions concerning its best management. Farmers or enterprise managers will sometimes want just the optimal configuration, but more commonly also need the comparison between alternate strategies which produce similar ultimate economic performance. Policy-makers (including politicians) may just wish to know which strategy to recommend as 'the best'.

One major consideration is the selection of just which of the modelled variables to concentrate on and optimise, and this depends on the scale and type of system being studied. For regional or larger systems, overall economic performance is typically used. However, this 'strictly mercenary' approach can ignore changes in infrastructure which may be detrimental to society, such as closing down facilities which currently provide local employment, or forcing boats out of a fishery without considering just where they are supposed to go (Mardle and Pascoe 2000).

At the farm or enterprise level, total farm production, or the gross value of production, have been used - for example, the milkfat production of a dairy farm (Hart et al. 1998). However, it is often the case that maximal production may only be achieved with excessively expensive inputs. Hence, some economic measure such as gross margin (the value of produce sold, less the costs of the variable input options) is usually considered. A number of variations are possible here - fixed costs may or may not be included, the effects of taxation can be factored in if required, and interest rates are usually included in dynamic problems. All this, however, still ignores any non-costable benefits of alternate options. Some of these can be included, for example paying the farmer a nominal hourly wage will pick up any increases in leisure time from improved management. However, other non-cash benefits can be difficult to include, for example the decreased use of pesticides can be advantageous environmentally, altruistically (as this delays the potential development of resistant strains of pests), and from a marketing perspective (it both lowers the chances of produce being condemned because of contamination, and gains a market advantage under the 'clean, green' image). Risk is also an important consideration with these systems (Feinerman et al. 1989), as maximal profitability is often achieved by driving the system harder, with increased chance of a biological crash. Utility functions which can incorporate varying levels of risk-averse behaviour can be used (Schilizzi and Kingwell 1999), thus allowing producers to adopt either a regret-minimisation or a risk-minimisation strategy (Paudel et al. 2000).

Despite these developments, some multiple-objective problems remain where a utility function approach cannot be used. Examples include cropping systems where both soil loss and profitability are important (and possibly differentially so between producers), and controlled breeding programs where genetic gain is to be maximised but level of inbreeding minimised (Meszaros et al. 1999). Frequently, these 'conflicting' objectives are strongly negatively correlated. If they cannot be appropriately weighted to form a combined single utility value, decision makers then require a curve or frontier, which portrays the trade-off between these key outcomes. Here, each identified solution is required to be Pareto-optimal or nondominated (Fonesca

and Fleming 1997). The traditional approach to obtaining this curve is to set one of these (such as maximum allowable soil loss) at a range of fixed levels, and then for each of these optimise for the other outcome (for example, profit) (Ritzel and Eheart 1994). More complicated methods, such as using Pareto rankings (where all nondominated solutions share the top rank), or scoring inversely according to the number of dominating solutions (Fonesca and Fleming 1997), are also available. Here, evolutionary algorithms can be used to good effect, as they naturally maintain a population of solutions. Over the time-frame of the optimisation, these should migrate onto the (optimal) Pareto frontier.

The second major use of optimisation in systems research is in the tuning of endogenous or internal model parameters. Here, the theoretical framework of the model has previously been defined, and the researcher needs to estimate parameter values to tune the model so that it best matches available data. These problems can arise when developing models, or applying existing ones to new environments, varieties, soil types, etc. The process is similar to non-linear statistical fitting, which cannot be used here because a model is not a single function. For model tuning, the choice of which variable to optimise is more straight-forward, and is usually taken as a summed measure of the difference between the observed and the modelled values. These include the sum of the absolute deviances (Goggos and King 2000), the residual sum of squares (Hendrickson et al. 1988, Hammer et al. 1993, Olsen et al. 1993, Franchini 1996, Franchini et al. 1998) or its log (Campbell et al. 1998), the residual variance (Wang 1991) or mean square (Cho and Lee 2000), and the scaled metric distance (Pabico et al. 1999). The independent variables are the model's internal parameters, which when optimised represent the tuned model. As with correlated parameters in nonlinear regression, various combinations of model values can contribute a similar degree of fit to the output results, implying potential multiple optima for the optimisation.

Having decided just which variable to optimise, most users require a robust 'black-box' optimisation algorithm which can be tagged onto and integrated with the systems model, and which will reliably return the globally optimal solution for any given scenario. Computationally, this relationship is often the other way around - the optimisation routine (of which there are many, as will be illustrated later) drives the whole process, and the model is coded in with this (often as a subroutine). The exact mechanics here depend on the computing platform and the code used in the model and the optimisation algorithm, as will be discussed in Chapter 3. More recently, some optimisers have also become available within major software packages, such as MATLAB and Microsoft Excel.

3. PROBLEMS

Simulation models in general, and agricultural systems in particular, constitute a class of applications which cause problems for many of the available optimisation techniques. Across the range of studied systems, these problems include the following.

1. No derivative functions are available. A number of optimisation algorithms (specifically, those methods which iterate using gradients) require first and second derivatives. Except in the simplest case of a single-function model (which presumably could be directly optimised), simulation models cannot be analytically differentiated. If derivatives are required for the optimisation, they must be approximated numerically, which introduces another possible source of error.
2. Practical constraints on the input options. Whilst optimisation algorithms are generally free to trial unlimited values of the independent variables, in practice these variables (which generally represent the management inputs into the system) have practical constraints. For example, it is impossible to apply negative amounts of irrigation or fertiliser to a paddock, and these and other inputs would also have realistic upper limits. Some optimisation methods (in particular, evolutionary algorithms) cater for this restriction in the 'minimum to maximum' coding of the problem, however others can requiring the extra complications of transformations of these variables, penalty function methods, or the use of constrained optimisation methods.
3. Shape of the response surface. Model output surfaces (mostly in higher dimensions, so not conducive to visualisation) are rarely smooth and convex (Fogel 1995a), as these complex systems are prone to instability (Woodward 1998). Economic results can range from 'bumpy' to 'almost chaotic' (Gleick 1987), depending on the problem. The response surface can have cliffs and discontinuities when the system is pushed too far – for example, if a grazing system is overstocked, it will collapse (both biologically and economically).
4. Multiple local optima. In agricultural systems these are quite common, resulting from alternate management strategies (often quite different in nature) producing similar economic consequences. For example, particular management strategies using low, intermediate, or high levels (and thus costs) of inputs such as fertiliser, irrigation and supplements may all be locally optimal, but these may (or may not) be similar in overall economic terms.
5. Size and complexity of the problem space. By definition, a useful simulation model includes all of the key variables and pathways of the targeted system. In non-trivial agricultural studies, the range of possible

management decisions is generally large. For example, dairy farmers have a multitude of available options, including pasture management strategies (areas and species of pastures and fodder crops, along with irrigation and fertilizer usage on these, and grazing patterns over time and space), supplementary feeds (types, levels and timing), herd size and balance (targeted numbers of breeders, heifers and bulls, and age structure), the use of artificial insemination, which breeds to use, calving patterns, culling policies, pest management (on pastures as well as animals), marketing strategies, infrastructure development (for example, shade areas to alleviate heat stress), and any diversification schemes (for example, farm tourism or aquaculture of crustaceans in the effluent ponds). Each property manager must evaluate the particular farm's resources, and then select a balance of these options to 'best' manage the farm enterprise. From a modelling perspective, many of these options have various biological or managerial constraints which must be taken into account. Constructing a valid simulation model capable of accounting for their individual and combined effects is a difficult task, as is finding the optimal strategy for these systems (Meadows and Robinson 1985).

6. Epistasis (interacting effects). In practice, many of the input options tend to interact strongly, whereas some of the optimisation methods rely on these being independent. This property can cause major problems for a number of optimisation methods (Mayer et al. 1998a). In agriculture, documented examples of this effect include the interaction between stocking rate and calving time in a beef herd in Victoria (Späth et al. 1984), the genotype by environment effect in the African grazing industry (Richardson and Hahn 1994), and the interaction between fire and grazing management in northern Australia (Liedloff et al. 1999).
7. Time paths and their dynamic nature. Natural systems do not generally exist in equilibria; rather they tend to be dynamic and/or transient (Onstad 1988). For any given generic system, a simulation model may have been used to identify the optimal static or steady-state configuration for the farm, however each farm will be starting with a different infrastructure and herd composition. Whilst the targeted situation is thus known, the direction and rate of progress towards this optimal state will vary with individual farm circumstances. The optimal solution over time for each farming system has to also include projected or potential changes in market returns, interest rates, taxation and level of debt. Also critical for many agricultural enterprises is the projected assumptions regarding weather patterns, including potential climate change scenarios and natural disasters such as cyclones, floods and (in particular) droughts

(McKeon et al. 1990). Nominated or historical climatic patterns can be used here.

8. Spatial heterogeneity. As the scale of the model grows, further complications also arise. In the central and northern regions of Australia, the average property size is 370 000 ha (O'Rourke et al. 1992). In these situations, spatial variability of soils and vegetation within properties becomes important, and needs to be taken into account (Hill et al. 1996, Mallawaarachchi et al. 1996). When scaling variables up to larger areas, aggregation errors can accumulate (Hansen and Jones 2000). For regional, state or national models, spatial random variability in a number of additional factors (such as weather, pest and disease incidence, and a range of animal performance parameters) causes further problems for modellers (Shaw and Findlay 1990). Also, at this scale the various sub-regions of the model tend to interact, for example a natural disaster in one region can cause a future drain in animal numbers from others, as well as immediately influencing the marketing structure and other components of the overall system.
9. Biological and system variability. Across all model scales, one of the major problems is caused by variability. This occurs in animal intake and performance (Fisher and Baumont 1994), meteorological events and rainfall distribution (Feinerman et al. 1989), commodity prices (Griffith and Piggott 1994, Cacho and Simmons 1999), and even farmer behaviour (Lazarus and Dixon 1984), including the degree of control of pests, and adoption of new technologies. For any system, the key processes are rarely known with any certainty, and confidence limits on these estimated relationships tend to be comparatively wide. Deterministic models attempt to simplify this problem by using the mean rates of each of these variables, and assuming any effect of variation will average itself out. However, their overall interacting effects can only realistically be simulated by modelling each with an appropriate distribution. In common with other disciplines, stochastic processes and models can be used to cater for variability. Here, multiple runs of the stochastic simulations contribute probability distributions of outcomes to be evaluated and compared, using stochastic dominance theory (Anderson et al. 1977), or by considering the average utility of multiple Monte Carlo runs (Day and Sparling 1977). With some exceptions (Mayer et al. 1994b, Buxton and Stafford-Smith 1996, Cacho and Simmons 1999), these methods are yet to be widely used in agricultural modelling.

Chapter 3

APPLICATION OF EVOLUTIONARY ALGORITHMS TO MODELS

Different forms of evolutionary algorithms have been developed for optimisation, in a number of scientific disciplines. The historical developments of the major underlying types are outlined, along with their current consolidation into a more standard and generic methodology. Published applications of the whole range of evolutionary algorithm types to the study of agricultural systems are listed (Appendix 1). These studies are categorised by application type, and the different forms of evolutionary algorithms used here are discussed. Finally, the different methods of interfacing the systems models with the chosen evolutionary algorithms are outlined and illustrated, with consideration given to problems and practical solutions.

1. FORMS OF EVOLUTIONARY ALGORITHMS

In a research equivalent of parallel evolution, different forms of evolutionary algorithms have been developed to practical fruition over the past few decades. The two most commonly used with agricultural and general models are genetic algorithms, primarily researched in the USA and English-speaking countries, and evolution strategies, which were largely developed in Germany (Hinterding et al. 1995). Other principal forms are evolutionary programming (Fogel 1995b) and genetic programming (Koza 1994), along with a range of evolutionary methods which are more specific to particular problem areas (Saloman 1996, Bäck et al. 1997a). The separate evolutionary disciplines first began communicating in the early 1990s (Bäck et al. 1997c), and since have swapped successful traits and strategies.

The generic term of evolutionary algorithms is now commonly used to cover this whole range of these methods. For the two types that cover most historical applications in the agricultural systems area, namely genetic algorithms and evolution strategies, summarised histories and general methodologies are as follows.

GENETIC ALGORITHMS

Based on the natural selection theories of John Holland, genetic algorithms have gained note with popular press articles such as Radcliffe and Wilson (1990) and Wayner (1991). Around the same time some more thorough texts appeared (Goldberg 1989, Davis 1991b), along with biennial international conferences on genetic algorithms being held. In practice, genetic algorithms have proven to be very targeted and efficient. In New Scientist, Radcliffe and Wilson (1990) reported 'good' solutions after only 10^5 trials in a system where complete evaluation was of the order of 10^{180}.

As their name implies, genetic algorithms are based on the biological concept of genetic reproduction, with successive gains being made by parallelling the process of natural evolution. This optimisation process takes the independent variables of the targeted problem and converts them into a genetic representation (usually binary). Here, grey coding is recommended as being more efficient than direct binary coding (Schaffer et al. 1989, Hinterding et al. 1995), as this guards against Hamming cliffs. A population of such individuals is obtained through random or targeted processes, with these forming the 'parents' of the next generation. Frequency of parenthood is directly related to fitness, which is taken as the value of the resultant dependant variable being optimised. So, like evolution, the successful individuals pass on their genes (attributes) more frequently.

The basic operation of genetic algorithms mimics sexual reproduction between two selected individuals, where sequences of genetic code are crossed and mixed to produce 'children' (the next generation), which are likely to be different from (and hopefully superior to) the parents. Over generations, this process tends to combine the more successful traits, and generally improve the fitness of the population. Low-level random mutation is also introduced to parallel nature, and this rediscovers 'lost' genes which may prove beneficial, as well as assisting in searching across the hyperspace. Over time, population structures tend to congregate around one (or a number of) optimal solutions. As with all current optimisation methods, there is no guarantee of finding the global optimum. Intuitively, the searching, mixing nature of recombination combined with the random element of mutation offers a relatively thorough coverage, even with extremely variable systems

(Klimasauskas 1992). The use of a number of random restarts remains good insurance though, and this has been shown to be beneficial in practice (Wayner 1991).

Under the initial genetic algorithm scheme, the variables to be optimised (i.e., the discrete management options of the simulation model) are mapped directly to binary genetic codings. If the number of discrete levels is not equal to 2^n (where n = 1, 2, ...), then null levels can be used, or the 'probable best' values of these options allocated more than once. The method of converting the continuous independent variables to discrete binary codings introduces both advantages and disadvantages for this method. On the positive side, bound constraints are implicit, as values are mapped onto a 'lowest to highest' defined range. If there is no realistic bound in either the upper or lower direction (i.e., possible values can range out to infinity), then a prior transformation such as the exponential may be used to stretch the range of values.

This discreteness of the representation of management options by binary genes also introduces disadvantages. Unlike the continuous optimisation methods, genetic algorithms cannot converge exactly onto an optimum, only to the nearest defined combination of input parameters. This problem can be alleviated by allowing more levels in each of these variables, however, this increases the length of the genes and thus the computational time to convergence. A safe approach, from a practical point of view, is to allow as many discrete levels as are realistically required in the real-world system that is being modelled.

One further disadvantage of current genetic algorithms is their lack of a defined termination method. Typically, genetic algorithms are run for a fixed number of generations or iterations, or until the population members have largely converged (within user-defined specifications) to a single solution. However, in multiple-optimal systems Goldberg (1987) shows that genetic algorithms are capable of the parallel investigation of a number of function peaks, including minor sub-populations around the lower optima. In these situations, the whole population will never converge to either a single area of the hyperspace, or to within any set difference between function values, so automatic termination methods would be difficult to implement. The safest option is to set the number of iterations at some high value where the algorithm has had sufficient time to search, cross-breed and mutate, so further improvement is unlikely to occur. This 'high value' tends to be problem-specific, and is largely determined by trial and error. In practice, it involves running some exploratory optimisations to excessive levels, to decide the appropriate number of runs.

As with any optimisation technique, researchers have trialed a range of operational parameters, adaptations and improvements. In general, optimal

settings appear to be problem-dependant, but robust values have been shown to work well across a variety of problems (Davis 1991b). The more important parameters and options include the coding of the problem's variables into an appropriate genetic representation, the choice of the population size, the method of selecting parents, the strategy for replacing population members, and the types and rates of mutation and recombination to be used. These will be discussed in Chapter 6.

Other adaptations and improvements have been proposed and trialed, including the much-discussed inversion procedure. Suggested as essential in early genetic algorithms, this methodological overhead did not prove useful or advantageous in practical applications (Davis 1991b), and has largely fallen by the way-side. However, it may yet become important as the number of genes increases (Davis 1991a).

Other developments which have met with mixed results include

- diploid (doubled-stringed) genes, which then require dominance/recessive genetic rules.
- restrictions on potential matings to reduce levels of 'inbreeding'.
- 'bit climbing' (Ackley 1987), which parallels the sequential one-dimensional hill-climbing methods.
- use of introns, which are segments inserted between coded operational bits (Angeline and Fogel 1997).

Some of these have demonstrated improved performance, but only in problem-specific areas. Goldberg (1989) suggests this is because the basic genetic algorithms work so well. Whilst Michalewicz (1996) suggests that a thorough evaluation of operational parameters to guarantee reasonable performance is required, Horton (1996) found that genetic algorithms perform well across quite a wide range of crossover and mutation probabilities. The use of genetic algorithms on real-world problems is becoming more widespread. In a wide range of practical applications, they are allowing the better solution of larger and more complex problems.

EVOLUTION STRATEGIES

The basic methods now collectively termed evolution strategies were developed in Berlin during the 1960s (Michalewicz 1996). Evolution strategies use real-number representation of the input values, so there is one 'gene' representing each modelled management option. Mutation is the key evolutionary operator, and was initially the only operation used with a population of one. Each 'offspring' is created by the random Gaussian mutation of each gene, according to a vector of mutation variances which itself evolves over time (Bäck and Schwefel 1993). Hence, the number of

genes is effectively doubled - for each input option to be optimised, one gene carries its value, and another its current mutation variance (Michalewicz 1996).

In evolution strategies, the fitness of each 'offspring' is judged in the usual way, namely the modelled value of the objective function using that member's trial combination of input variables. Whilst the mutation variances don't feature directly in this calculation, it does allows the more successful combinations of mutation variances to be 'dragged along' into the next generation, as they will be the ones generating the model input values which produce the best values of the objective function. These variances are initially larger to facilitate searching, and over time narrow down to near zero (and effectively fine-tune the solution) as the genes converge to their optimal value (Michalewicz 1996). Some applications also carry a third vector of covariances or correlations between the respective mutation amounts, allowing directional searching which is then optimised over time.

The single-parent and single-offspring evolution strategy was adapted (Bäck and Schwefel 1993) to include multiple parents (with the number of these usually represented by μ), and multiple offspring (λ). Selection of parents for reproduction is usually random, and selection pressure is introduced by the offspring being retained or discarded on a deterministic assessment of fitness (Bäck and Schwefel 1993). Two separate strategies are available - the (μ,λ) evolution strategy, where the parents are replaced by the best offspring at each generation, and the $(\mu+\lambda)$ evolution strategy, where the combined parents and offspring compete to be amongst the best μ individuals used as parents in the next generation. The former strategy has proven useful for noisy functions and problems where the optimum is non-stationary (Michalewicz 1996). The $(\mu+\lambda)$ evolution strategy is somewhat equivalent to the elitist strategy of genetic algorithms (Mühlenbein and Schlierkamp-Voosen 1994), in that the best strategies are preserved. This appears more appropriate to the optimisation of deterministic agricultural models.

Regarding the balance between μ and λ, Fogel (1995b) showed that the $(1+\lambda)$ evolution strategy has a logarithmic increase in the rate of convergence over the (1+1) evolution strategy, and Mühlenbein and Schlierkamp-Voosen (1994) demonstrated a speed-up as λ was increased. Bäck and Schwefel (1993) found an optimal ratio of μ:λ of about 1:7 for their types of problems.

CONSOLIDATION OF EVOLUTIONARY ALGORITHMS

As with other competing methodologies, some early studies investigated the relative practical performance of genetic algorithms versus evolution strategies. On test functions, Bäck and Schwefel (1993) showed evolution strategies identified better optima, and at faster rates, than binary genetic algorithms. However, Keane (1996) found the opposite when comparing a sophisticated binary genetic algorithm against evolution strategies on difficult test functions. In the practical optimisation of laminate designs, Le Riche et al. (1995) found these two methods performed similarly. Hinterding et al. (1995) reported mixed results - genetic algorithms proved superior on discontinuous and multiple-optima test functions, whilst evolution strategies were better with lower-dimensional and smooth-type functions. Currently, though, genetic algorithms and evolution strategies may be viewed more as collaborators than competitors. As mentioned, the past few years have seen these two methods effectively merge into an even more powerful class of methods, now generally termed evolutionary algorithms.

This generic term also includes other forms of evolutionary computation. Whilst genetic programming (Koza 1994) was initially developed for self-evolving computer programs, Whigham (1999) used this framework to develop a plankton dynamics model in freshwater lakes. Results were superior to both a theoretical model and a neural network. Operational parameters included a large population size of 500, a crossover rate of 0.9 and a mutation rate of 0.05, which identify their method as an evolutionary algorithm.

Evolutionary programming (Fogel 1995b) was initially involved with the development of artificial intelligence (Bäck et al. 1997c), and has since seen wider application. It does not include recombination (Spears 2000), and is largely driven by selection as a culling force, plus mutation (Bäck and Schwefel 1993, Fogel 1995b). In a comparison using test functions (Bäck and Schwefel 1993), applications of evolutionary programming, evolution strategies and genetic algorithms gave mixed performance – the best algorithm depended on the form of the function being optimised.

The practical integration of these different forms of evolutionary algorithms has resulted in a number of changes and enhancements to their respective basic formats. Most importantly, the adoption of real-value rather than binary genes is now more common, to more closely match the continuous nature of the variables of most problems. However, binary representation can still be used, particularly if the model's options align with discrete levels. The adoption of real-value coding opens up a range of

possible recombination and mutation operators, as will be outlined in Chapter 6.

Another important consideration is whether to code mutation variances as a second suite of genes, as per the evolution strategies approach. If adopted, this effectively doubles the problem size. The alternative here is to use static or dynamic mutation rates across all variables, in line with the genetic algorithms method.

In evolutionary algorithms, the role and importance of the major operators remains open. Each offers a balance between exploitation (using existing material to best benefit) and exploration (to adequately search the feasible space). Traditionally, genetic algorithms were driven by recombination, with mutation only a background operation (typically having a probability of 0.01 or lower), whereas evolution strategies initially only used mutation. The optimal balance between recombination and mutation can vary, and is likely to be problem-specific (Michalewicz 1996). Using test functions, Hinterding et al. (1995) found the best combination was lower crossover rates (0.1 to 0.4) with moderate mutation (up to 5/N, N being the number of genes). On a transportation problem, the optimal combinations were crossover rates of 0.05 to 0.25 with mutation probabilities around 0.2 to 0.4 (Michalewicz 1996). Significantly, in this transportation study, crossover rates of zero (thus leaving mutation as the only operator, in the style of evolution strategies) proved inferior, as was also reported by Bäck and Schwefel (1993). A number of studies have shown mutation and recombination to be advantageous to each other (Fogel 1995b).

An early practical example of a 'combination' evolutionary algorithm was the breeder genetic algorithm of Mühlenbein and Schlierkamp-Voosen (1993, 1994). More recent applications from the agricultural systems field include Polheim and Heißner (1997), Arias et al. (1998), Campbell et al. (1998), Mayer et al. (1999b), Meszaros et al. (1999), and Mardle and Pascoe (2000). As Hammel (1997) points out, evolutionary algorithms (of whatever form) are ideally suited to the optimisation of simulation or systems models. They are inherently parallel, have proven to be robust across a wide range of parameter settings, and are amongst the most efficient of the available optimisation methods.

2. AGRICULTURAL MODEL APPLICATIONS

Compared with other research areas, the agricultural disciplines have proved to be relatively slow in the uptake of evolutionary algorithms. In the more recent years, however, the trickle of evolutionary algorithm applications to various areas within the agricultural sciences has turned more

into a flood. Appendix 1 lists those found in the literature to date, with more appearing regularly. Overall, applications of genetic algorithms (originally only binary, but more lately including real-value codings) dominate, and few evolution strategies have been published in this field. The width and depth of these applications illustrates the different ways that evolutionary algorithms can be used in agricultural systems, in terms of the types and sizes of the systems modelled, the different forms of the objective functions, and the operational parameters of the various evolutionary algorithms (as will be discussed in Chapter 6). These applications of Appendix 1 can be categorised into a number of common themes.

GREENHOUSE PRODUCTION

The problem of maintaining optimal greenhouse conditions for the production of horticulture or floriculture is the topic for a number of studies. The first to appear was Annevelink (1992), where a spatial and temporal allocation of crop cycles was solved with a binary genetic algorithm, after initial linear and dynamic programming approaches proved inadequate. Morimoto and Hashimoto (1996) also used a binary genetic algorithm to determine optimal greenhouse tomato production, having modelled the crop growth and nutrient concentrations at critical physiological times.

The remaining four greenhouse studies all modelled the heating costs and temperature controls of these systems. Polheim and Heißner (1997) included ventilation and CO_2 enrichment as options in a complex 51-variable problem, which was solved via a combination of a real-value genetic algorithm to conduct the global aspects of the search, and an evolution strategy to fine-tune the solution near the optimum. Arias et al. (1998) also used a real-value genetic algorithm, noting that this representation aligned better with the greenhouse control parameters. In this system, the genetic algorithm set the targeted values for the real-world neural network and fuzzy controllers to then achieve. Goggos and King (2000) used a hybrid (of a binary genetic algorithm plus a hill-climber) to minimise deviations from the targeted temperature schedule, and Husmann and Tantau (2001) utilised a binary genetic algorithm with a greenhouse model which incorporated heat storage, boiler configuration and fuel type. This study reported near-optimal results after only 10 generations, or a total of 300 runs of the simulation model, indicating the relative efficiency of this evolutionary algorithm.

FARM MANAGEMENT

One of the most logical, and most used, applications of evolutionary algorithms to agricultural systems is in the optimisation of whole-farm or property systems. These are typically complex and multi-dimensional models, which cannot easily be optimised. A model of a dairy farm in the sub-tropics of Australia was optimised using the GENESIS binary genetic algorithm (Mayer et al. 1995, 1996a), and incorporated management decisions covering pasture types and areas, applications of fertiliser and irrigation, grazing management, calving patterns and the use of supplementary feeds. A similar study, modelling a pastoral dairy farm of New Zealand (Hart et al. 1998), found the optimal management regime after only 15 000 model runs. Parmar et al. (1996) optimised the number and type of machinery selected for peanut farm management in Georgia (USA), and Parsons (1998) optimised the spatial and temporal harvesting plans for silage on a farm system in the UK. Both these studies used binary genetic algorithms. Barioni et al. (1999a) modelled the key management decisions (paddock grazing rotation, fertiliser applications, lamb drafting policy, and supplementation) over time, for a New Zealand sheep farm. The optimal management policy found proved to be consistent with the recommended industry best practice for this system.

Different combinations of farm investment strategies were stochastically modelled and optimised by Cacho and Simmons (1999), resulting in cumulative distributions to be interpreted. They used a binary genetic algorithm, but suggested that real-value coding may perform better. In the latest farm-level study to date, a beef property in northern Australia was modelled - firstly considering 40 annual trading (i.e., buying and selling) decisions (Mayer et al. 1999a), and then expanding the model to include within-year details and stocking pressure, resulting in 70 dimensions to be optimised (Mayer et al. 1999b, 2001). This series of studies contrasted the performance of both binary and real-value genetic algorithms with evolution strategies. The genetic algorithm versions were found to perform similarly, but were superior (in terms of the rates of convergence) to the optimisations using evolution strategies. These results were attributed to the latter's requirement of an extra gene (to carry each option's mutation variance), for each of the 70 management options.

REGIONAL MODELS

The next level of complexity up from the individual property is a multi-farm or regional model. Traditionally, many of these types of systems have

been modelled using linear or mathematical programming, as until recently these were the only practical method of optimisation. Kuo et al. (2000) modelled irrigation usage and crop types and areas in a number of farming basins in Utah (USA), and found a binary genetic algorithm to be superior to hill climbing and simulated annealing. Mardle and Pascoe (2000) developed a complex model of the English Channel fisheries system, with 876 variables covering the fleet size, configuration and targeted species by seasons. Despite a run-time of several hours, the real-value and integer genetic algorithm effectively solved this problem, for which a linear programming approach was found to be infeasible. Despite these few successes at the regional level, the application of evolutionary algorithms to national or larger systems is yet to be noted.

BREEDING PROGRAMS

An interesting application of the 'artificial genetics' of evolutionary algorithms is to the modelled genetic gains in breeding programs, in both the animal and plant sciences. Genetic algorithms have been used to effect here because they can model the dynamic and temporal nature of these problems, unlike some of the static methods (Hayes et al. 1997). At the farm-level, a binary genetic algorithm was used successfully to analyse mate selection for the pairing of individual animals (Hayes et al. 1997). For the national sheep breeding industry of Australia, Horton (1996) investigated the optimal arrangement of the number of tiers, flock sizes and age structures, exchange rates, and selection mechanisms. The binary genetic algorithm used here identified optimal solutions which hill-climbing methods had repeatedly failed to find. Similarly, Meszaros et al. (1999) also investigated large-population animal breeding strategies (maximising genetic gain, penalised for levels of inbreeding), using a real-value genetic algorithm. In the plant sciences, Verryn and Roux (1998) found a genetic algorithm to be superior to the statistical method of best linear unbiased predictors, for selection in a South African forestry breeding program.

COMBINATORIAL OPTIMISATION

Despite then suggested superiority of alternate optimisation methods (such as the tabu search strategy) on these types of problems, evolutionary algorithms have been used to effect here. The costs of a water distribution network in Mexico were minimised using a binary genetic algorithm (Mariano 1998). On this complex case study, which would take three

million years of computation to solve by complete enumeration, the genetic algorithm outperformed two heuristic methods, by 4 and 19% respectively. The dispatch processes of a nursery were modelled in Kozan (1999). On this travelling-salesman type problem with 21 nodes, a discrete-value genetic algorithm outperformed a number of problem-specific heuristics. The routing of an autonomous orchard sprayer was optimised in Cho and Lee (2000), using an integer genetic algorithm which was integrated with a global positioning system and real-time operational software to control the sprayer unit. In this study, the simulations indicated a 68% improvement on the baseline, but in the real world this was very much reduced, due to tyre slip and the response time of the hydraulics. However, these faults cannot be attributed to shortcomings of the evolutionary algorithm, but rather to inadequate engineering.

The spatial and temporal harvesting schedules of forest stands also form a version of combinatorial optimisation. Moore et al. (2000) found optimal harvesting decisions after only 200 generations, using a trinary genetic algorithm (i.e., each coded option took one of three possible values). In a similar silviculture application, Lu and Eriksson (2000) used a two-stage binary genetic algorithm - firstly an exploratory search using a coarse-grid of 625 spatial cells, followed by a fine-tuning version with 10 000 cells of trees.

STATISTICAL APPLICATIONS

A novel application of a genetic algorithm to experimental design is outlined in Davies et al. (2000). Here, it is not a systems model which is being optimised; rather, a series of agricultural experiments, investigating the combinations of eight additives to aid silage production. With two weeks being required to physically run and analyse each trial, only five generations were possible, each having a population of 50 (trialed combinations of additives). Of the three elite solutions at the end of these experiments, two were shown to outperform the industry standard formulations.

The fitting of model parameters to observed data form the majority of applications of evolutionary algorithms to statistical-type problems, using binary (Wang 1991, Pabico et al. 1999) or real-value (Franchini 1996, Franchini et al. 1998) genetic algorithms, or a combination of both types (Campbell et al. 1998). For these studies, a range of statistical measures are minimised (Appendix 1), all of which measure the degree of discrepancy between the observed and predicted values. In general, these were successful - Wang (1991) reported near-optimal results after only 4 000 runs, for a 7-parameter model with a search-space of the order of 10^{21}. A far more

difficult problem set of 42 parameters was fitted for a model of the degradation of pesticide residues in wool (Campbell et al. 1998). Given the nature of this model, it is likely that these parameters were largely independent of each other (i.e., there was little epistasis), which makes the problem somewhat easier to optimise.

Pabico et al. (1999) found genetic algorithms to be superior to hill-climbing methods in determining cultivar coefficients for crop models (13 parameters and a search-space around 10^{32}). Conversely, on a hydrological model, Franchini (1996) required a local search heuristic to be hybridised with a real-value genetic algorithm before convergence was achieved, and Franchini et al. (1998) showed there was no 'globally superior' algorithm for the models listed. In a number of these case studies, the genetic algorithm was outperformed by heuristic methods. The size of these examples (11-13 parameters with search-spaces around 10^{23}-10^{33}) should not contribute to this lack of convergence problem - as outlined above, larger models have been successfully fitted. The nature of this model (Franchini 1996, Franchini et al. 1998) indicates that it is more likely to have interacting parameters, which may have caused these problems. The application of evolutionary algorithms to the statistical fitting of nonlinear regression problems, as was done with simulated annealing (Mayer et al. 1996b), appears a logical extension into this field. However, this has yet to be noted in the literature, and genetic or evolutionary algorithms are yet to be incorporated in the major statistical software packages or libraries.

3. MODEL / OPTIMISATION INTERFACE

Users wishing to optimise their systems model using an evolutionary algorithm need to integrate these two programs. Obviously, this is easiest if both are written in the same language, but of late compiled versions can effectively be combined. For the interface, it is common to have the evolutionary algorithm as the 'main' (driving) program, with the model being inserted as a subroutine or function. Each call to this subroutine is a trial run of the model, as the main program conducts its search for the optimal combination of model inputs.

The only common methodological feature between the wide range of applications of evolutionary algorithms to agricultural systems is that there appears to be no common methodology. This may be due in part to the differing types of problems being modelled, but is more likely a result of the wide choice of algorithm types and sources. Given that no one of these is globally superior, most users find their adopted method works quite well, even if it usually requires some minor tuning or parameter adjustment.

Many users code up their own version of their desired evolutionary algorithm. Hart et al. (1998) and Mariano (1998) based their programs on the 'simple genetic algorithm' (SGA) of Goldberg (1989). Fortran versions of evolutionary algorithms were coded in Parmar et al. (1996), Meszaros et al. (1999), and Lu and Eriksson (2000), while Horton (1996) used Pascal, and Verryn and Roux (1998) used Clipper. Regardless of the computer language used, this approach involves complex coding and extensive testing, to ensure that the programs are actually doing what they're supposed to. This approach is fine in that the user has direct control over the methodology, and if a learning experience is required. However, practical systems modellers usually require efficient, proven algorithms that they can simply hook up with their model.

The range of 'off-the-shelf' evolutionary algorithms, both shareware and purchasable, is well exemplified by the comprehensive compilation listed on http://www.aic.nrl.navy.mil/galist/src. Whilst many of these are promoted as 'plug-in-and-go' black box optimisers, some considerable effort is still required in coding the interface between the chosen algorithm and the user's model. Further time is often then spent in gaining an understanding of the package's various options, and getting the algorithm efficiently tuned and working with the problem at hand. Hence, few users will be familiar with more than one algorithm, as once they have become familiar with its particular foibles and advantages, they will then tend to reuse it for new problems.

Our research (Mayer et al. 1995, 1996a, 1999a, 1999b) has extensively used GENESIS (Grefenstette 1995), a binary genetic algorithm, with few problems. For real-value codings of the same models (Mayer et al. 1999b, 2001), we have also used Genial (Widell 1997). Whilst both these packages performed successfully, this is not necessarily an endorsement or recommendation for either. Most packages would probably work as well, as they are all based on the proven efficiencies of evolutionary algorithms. Polheim and Heißner (1997) used the MATLAB toolbox, and Woodward (1998) adopted Evolver, both with success.

The overall recommendation is thus to use one of the many available off-the-shelf evolutionary algorithms. These are generally easy to integrate and use, and most allow the different methodologies and parameter options that a user may wish to trial.

Chapter 4

APPLICATIONS OF ALTERNATE OPTIMISATION TECHNIQUES

This chapter covers a number of applied optimisation techniques which do not fall under the 'evolutionary algorithms' category. It gives a brief outline of some methods which proved to be only of limited use for systems models, and then overviews the generic types of applicable optimisation classes. This includes discussion on their respective methodologies, natural advantages, and problems and disadvantages, and how they have been applied to real-world systems in general, and agricultural systems models in particular. These respective classes of optimisation methods are gradient-type (hill-climbing) and its derivatives, direct search methods, simulated annealing, and the tabu search strategy. Hybrid methods, which incorporate more than one of these available families of techniques (as well as evolutionary algorithms), are also considered.

1. INTRODUCTION

When 'A new algorithm for optimisation' (Anonymous 1972) appeared in the journal 'Mathematical Programming', a great breakthrough had apparently been made. The paper outlined a relatively simple procedure which would simultaneously cater for nondifferentiable functions, integer variables and a nonconvex domain, whilst also having proven convergence at a very efficient rate. Unfortunately for researchers in this field, however, it proved on closer inspection to be a (mathematically well-disguised) send-up of procedures in this complex area - a fact that has been missed by a number of readers who have apparently taken it seriously (Polyak 1987).

Since the 1950s, studies in methods of optimising non-linear problems have increased markedly in many scientific disciplines (Gill et al. 1981). With the passing of each decade, advances in techniques and computing power have enabled the solution of problems of a size and complexity which previously could not be contemplated. For the generally applicable methods of optimisation, this area of applied mathematics remains a rapidly changing and evolving discipline. A number of suggested optimisation methods have proven to be too inefficient, impractical, or ill-suited to the general optimisation of systems models. These include -

1. Complete numerical (factorial) evaluation of the feasible space. Here, every level of each input option is tested in turn with each of the levels of all other factors, and the optimum taken as the best value obtained. If more precision is required, a finer grid can be used across the optimal region. Whilst a comprehensive evaluation method, it is unworkable with even moderately-sized problems, and the only agricultural literature examples are from models with a small or deliberately simplified search-space. Rodriguez et al. (1990) evaluated eleven stocking densities by nine time combinations in a grazing systems model of the southern USA, and Gates et al. (1994) considered discrete temperature settings for a controlled-environment piggery model. A seven-way factorial of a dairy genetic improvement model, with a total of 640 discrete combinations, was subjected to analysis of variance (Mayer et al. 1994b). McIvor and Monypenny (1995) conducted a two-way factorial design on their beef property model, and Buxton and Stafford-Smith (1996) compared five discrete rangeland management strategies. Paz et al. (1999) investigated a range of 21 rates of nitrogen, using the CERES-maize model.
2. Random (non-directional) search patterns. As may be expected, these tend to be very inefficient, and are seldom used in practice. In the optimisation of test functions and models, a number of case studies (Corana et al. 1987, Bramlette and Cusic 1989, Bramlette and Bouchard 1991, Davidor 1991, Syswerda 1991) have used random search methods, mostly only as benchmarks to demonstrate the superiority of the more targeted and efficient algorithms. However, recently this underlying principle (of randomness) has been brought back as a feature of some of the more robust global search methods.
3. Sequential one-dimensional optimisations. Here, each management option (or dimension) in the model is optimised in turn, either automatically by the algorithm, or 'manually' via the user during exploratory model runs (Thornton and McGregor 1988, Feinerman et al. 1989). These methods become unwieldy with even a moderate number of dimensions, and also suffer severely when dimensions have an interacting effect on the variable to be optimised (i.e., when the

dimensions are not independent). As has been shown, this is often the case with models of agricultural systems.

4. Approximate the surface by a smooth function or meta-model. Polynomial functions are generally favoured, as these can be solved easily to find the optimum, via standard mathematical methods (Bertsekas 1975). This approach was used in simulation of the reproductive performance of a beef herd (Notter and Johnson 1987, Notter and Johnson 1988). Whilst intuitively appealing, problems arise from the key assumptions of smoothness and symmetry of the response surface, and independence of the dimensions, as well as the potential lack of fit between the systems model output and the meta-model (Kleijnen 1987). Even if the response surface was smooth and unimodal, one of the gradient techniques would appear to be more useful, as these work directly with the model output values.
5. Branch and bound algorithm. In practice, this method is not well suited to the optimisation of general systems models. Being an exact method, it is only useful with smaller problems (Osman 1993). On larger models it can be markedly inefficient, and numerically degenerate to almost complete evaluation (Pardalos et al. 1995). For example, on an allocation problem for groundwater remediation, a branch and bound algorithm took over 10^7 model evaluations to identify an optimal solution that a genetic algorithm found in only 1 250 model runs (Wagner 1995). Similarly, Kozan (1999) showed that the branch and bound method failed on a 21-node nursery allocation problem - even after extensive computation, its best solution was 10% worse than that of a genetic algorithm. On a moderately-sized forestry management problem with 84 grids, Bos (1993) found the branch and bound method to be computationally intractable, and opted to use simulated annealing instead.
6. Neural networks. This forms a large discipline, primarily aimed at developing network models to predict data patterns. As such, neural networks themselves use many of the optimisation methods outlined in this monograph, to maximise their degree of fit. A range of allocation-type optimisation problems have been tackled by neural networks, including graph partitioning and map colouring (Ritter et al. 1992), and the application of neural networks and genetic algorithms to the travelling salesman problem (Murtagh 1994). On a protein-folding problem (via lattice partitioning), Rabow and Scheraga (1993) found neural networks to be more successful at finding the global optimum than a standard simulated annealing. Conversely, Peterson and Söderberg (1989) showed that simulated annealing found better solutions than neural networks, in graph partitioning (0.5% higher), and travelling

salesman problems (8 to 10% better). In developing a mathematical predictive model for the chlorophyll concentration of freshwater lakes, Whigham (1999) found the optimised mean square error of a genetic program to be 37.1, which was superior to that of a neural network, at 41.8. Overall, however, neural networks cannot be classified as a method which can be generally applied to the optimisation of systems models, and no examples of this have appeared in the agricultural systems literature.

2. GRADIENT-TYPE METHODS

The common property of this multitude of techniques is that they calculate or estimate the gradient and curvature of the response surface at each iteration, thus determining the direction and distance of the next jump. Successive iterations (hopefully) converge towards an optimum, as shown graphically in Hart et al. (1998). At each iteration many calculations are made in the vicinity of the estimated point, making these methods well suited to parallel computing (Bertsekas and Tsitsiklis 1989). For smooth, unimodal functions (or models which produce this type of response surface), these methods are amongst the most efficient available (Fletcher 1987). For a range of these problem types, efficient convergence can be proven (Dunn 1981).

FORMULATION

The mathematical derivation of these methods has been well documented (Bunday 1984, Fletcher 1987, Fryer and Greenman 1987, Polyak 1987). The shape of the response surface is estimated via Jacobian and Hessian matrices, which contain the first and second derivations respectively, for each dimension of the hyperspace. If these values cannot be directly calculated mathematically, they can usually be estimated with sufficient accuracy by finite differencing or quadratic approximation (Madsen 1975). This situation occurs with output from simulation models, which cannot be differentiated analytically.

The three necessary conditions for x^* to be a global maximum of f(x) are

$$\forall_x,\ f(x^*) - f(x) > 0,$$

$f'(x^*) = 0$, and

$f''(x^*) \leq 0$

Most gradient algorithms use the second and third of these conditions to determine the stopping point, which is then taken as the optimum because no further progress can be made in the immediate vicinity. Here, the first condition also holds for at least the local environment, but this is not guaranteed globally. This causes the most serious deficiency with these methods - whilst they efficiently track up-hill to the top of the surface, there could well be a higher peak elsewhere on the response surface. Any peak in the multi-dimensional hyper-surface will satisfy these conditions locally, and qualify as a local maximum. However, finding the absolute highest point, or global maximum, can prove difficult. If these methods fall into the basin of attraction (Grebogi et al. 1987) around a local maximum, they are trapped by it, and will then only find the global maximum if a jump allows an iteration to escape from this basin. The only way to deal with this shortcoming is to trial many different starting conditions, well spread across the feasible hyperspace. Each solution (maximum) should be evaluated, and the highest then adopted as the global maximum. However, in multi-dimensional problems with multiple local optima, even this solution may only give a reasonable to poor chance of finding the global maximum (Polyak 1987, Mayer et al. 1991).

Potential users are faced with a wide variety of available methods. In many cases, the best choice will depend on the type of problem being optimised. Some methods have been developed specifically for the efficient solution of particular problems, but these cannot necessarily be expected to work well outside their target area. Examples include the minimisation of sums of squares in nonlinear regression (Bard 1974), networking problems (Fisher et al. 1975), and geometric programming (Fletcher 1987, Fryer and Greenman 1987). If the problem can be formulated as one of these, the choice of an algorithm is clear, as these areas have been well investigated. When facing the optimisation of an unknown response surface of a simulation model, however, it seems more prudent to use one of the range of general gradient methods.

GENERAL METHODS

The simplest general methods are those which determine the direction of steepest ascent (Wolfe 1975). From the starting point, these methods

estimate the maximum positive gradient, and search in that direction. However, their performance can be somewhat erratic, as they are particularly prone to zig-zagging over successive iterations (Shor 1985), and have difficulty dealing with ravine-type functions (Polyak 1969). Powell (1964) outlines a method in which conjugate gradients are chosen. This adaptation has distinct advantages in performance over simple ascent methods (Radcliffe and Wilson 1990). Various computational schemes are available depending on the problem, including the well-known Fletcher and Reeves method (Fletcher 1987). These techniques have certain computational advantages over the Newton methods described below, and are often better suited to large problems.

Newton's method remains the corner-stone of a second family of techniques (Smale 1986), and this method is used as an introductory example in many of the texts. Newton's method is based on a quadratic model, which is derived from a truncated Taylor series expansion of f(x) (Fletcher 1987). Solution of this quadratic model (in each dimension, for multi-dimensional problems) provides the next point for iteration. For smooth-surfaced problems which are well approximated by quadratics, Newton's method generally converges rapidly to an optimum.

Methods which require the estimation of the inverse of the second derivative matrix are termed quasi-Newton, with performance generally similar to Newton's method (Polyak 1987). A range of improved formulations of the quasi-Newton have been proposed (Dennis and More 1977, Dembo et al. 1982), and tested in case studies. One generalisation of these is the Broyden family of formulae (Fletcher 1987), which combine the Davidon, Fletcher and Powell (DFP) method and the Broyden, Fletcher, Goldfarb and Shanno (BFGS) formula (Fletcher 1982, Powell 1982). Numerical case studies have demonstrated similar performance between the DFP and BFGS (Fletcher 1987). Whilst these adaptations have certain advantages for particular problems, rarely will the user know the best choice of the required formula parameters to optimise an unfamiliar problem.

For problems with a relatively high number of dimensions and a largely unknown response surface, either the general quasi-Newton or conjugate gradient methods appear most promising. For deterministic problems, Polyak (1987) concluded that the conjugate gradient method performed better than the quasi-Newton, which in turn was superior to a range of others. Conversely, Fletcher (1987) found that in practice the quasi-Newton was more robust than the conjugate gradient method, with resultant superior performance. Overall, there appears little difference between these when applied to the smooth, unimodal surfaces for which they were specifically developed. However, their performance on noisy, non-smooth real world

systems remains yet to be adequately tested (Polyak 1987), and may well be expected to be poor.

TRANSFORMATIONS

It is readily accepted that optimisation techniques require (at least) a scaling transformation if the various dimensions exhibit a range of scales (Fryer and Greenman 1987). Both the independent variables and the objective function should be scaled to approximately the same units, usually in the range 0 (or -1) to 1. Linear transformations are most commonly used to achieve this, but nonlinear transformations such as the square-root, power or log can be used if the particular variable displays the appropriate distribution.

Duality involves transforming the original formulae or problem into an alternate formulation which has some theoretical significance, or is easier to solve computationally (Fletcher 1987). For certain types of problems this can improve the efficiency of gradient methods (Fisher et al. 1975), but it is difficult to imagine an overall method which would be applicable to the general optimisation of systems model output.

Some of the more complex transformations include the space-variant developments, where dilation or contraction of the vector space is conducted (Balinski and Wolfe 1975). Shor (1985) outlines algorithms for space dilation along the gradient, which aim at widening the sometimes acute cones of ascent directions. Other techniques include relaxation methods (Camerini et al. 1975) and contraction mappings (Polyak 1987, Bertsekas and Tsitsiklis 1989). These transformation methods are somewhat specialised and suited to particular problem types, and as such appear to lack generality.

CONSTRAINED OPTIMISATION

Much has been documented on constrained problems, the constraints being either linear or nonlinear, and equality or inequality. Inequality constraints produce planes (of combinations of the variables) which intersect to define the feasible region, with combinations of values of the variables beyond these being infeasible. Equality constraints imply feasible planes through the hyperspace, which can then be searched efficiently by specialised techniques. In many agricultural models, inequality constraints are required to define physical ranges on the variables. Constraints involving more than one variable are less common, but still possible.

Fortunately, simple range (or bound) constraints are amongst the easiest for optimisation methods to cater for.

For bound linear or non-linear inequality constraints, the use of penalty function methods has shown great universality (Polyak 1987). These methods add some penalty value to the objective function at infeasible locations, forcing the methods back into the feasible region. Extreme penalties, where a nominally large value is added, result in barrier functions, where violations are immediately rejected. Whilst effective, these barrier methods produce a cliff along the constraint, which is where the global optimum may actually occur. The resultant lack of smoothness can cause serious problems with gradient methods. More useful are linear or nonlinear penalty functions, which increase with the degree of violation. Figure 1 shows examples, taken from the dairy model optimisations in Mayer et al. (1996a, 2000), of some of these commonly-used penalty function methods.

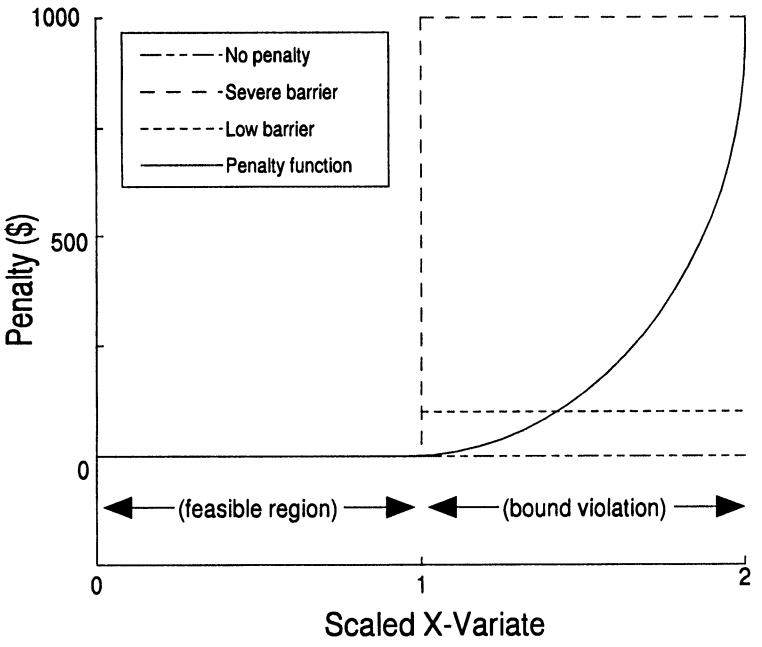

Figure 1. Penalty methods for violation (X>1) of a bounded variable.

In some algorithms, a preferable way of dealing with constraints is the feasible directions method (Fletcher 1987). Here, any non-feasible variable value is projected back onto the nearest active constraint. For simple bound

constraints, any value beyond a bound is merely set at that bound. For multi-variable constraints the implementation becomes more difficult, but this basic property is retained.

Probably the most efficient technique for handling constraints, however, is the method of feasible gradients (Polyak 1987). In this, active and near-active constraints are taken into account along with the surface gradient when determining the direction of the next iteration. Hence, the method rarely ventures out of the feasible region, but tends to track along any constraint which contains an optimal solution. This efficient heuristic is unique to the gradient-type optimisation techniques.

PROBLEMS

By their very nature and derivation, gradient-type methods have definite theoretical and practical shortcomings when applied to the optimisation of real-world system models. As outlined earlier, by design they converge to the (assumedly single) optimum of the system, so in practice they track up-hill to the closest maximum, which is unlikely (in multiple-optima systems) to be global. These methods retain no memory, and (unless lucky) cannot usually escape from local optima. The only exception here is the heavy ball method (Polyak 1987) which can 'roll across' relatively shallow local optima, but otherwise has displayed relatively poor performance.

The second major problem concerns the rough, sometimes almost fractal, nature of potential response surfaces. Polyak (1987) contends that most results of minimisation studies on smooth functions will not extend to non-smooth problems, and this point has been demonstrated with a complex dairy model (Hart et al. 1998). The cliffs and occasional discontinuities of these surfaces can cause havoc with derivative estimates. Also, noisy or stochastic systems can cause problems, as gradient methods tend to be very sensitive to errors or outliers.

AGRICULTURAL MODEL APPLICATIONS

Given the limitations of gradient-type methods, it is not surprising that relatively few applications have appeared in the agricultural literature. Roise (1990) used the conjugate directions method to optimise spatial models of forestry stands, however these models were only small to moderately sized. Barioni et al. (1999b) showed that a generalised reduced gradient algorithm repeatedly failed to find the optimal configuration of a feedlot diet formulation problem. Generally, gradient methods have more been used as

'the traditional benchmark', against which alternate optimisation methods are shown to be superior, as in Hendrickson et al. (1988), Mayer et al. (1989), Wang (1991), Mayer et al. (1996a), Hart et al. (1998), Parsons (1998), and Kuo et al. (2000).

3. DIRECT SEARCH ALGORITHMS

As the name implies, direct search methods use only function (or model) values, retaining a certain population of input combinations which are iteratively improved. By not attempting to estimate the slope or curvature of the response surface, they traditionally have been termed *ad hoc* methods, and perceived as generally inferior. In practice however, these methods have proved to be worthwhile, especially for finding the global rather than a local optimum.

One of the most commonly used methods is the simplex, proposed by Spendley et al. (1962) for the optimisation of factory production. Given a problem in n dimensions, a simplex is a collection of (n + 1) points or vertices, which (in the absence of linear dependencies) define a hypershape in the Euclidean space. In the literature, some confusion exists with the simplex solution method for linear programming (Lagarias et al. 1998), such that the general simplex optimisation method has sometimes been called the simplicial (Polyak 1987), or the polytope method (Gill et al. 1981). The original terminology is preferable, and has mainly been used in subsequent studies. The type of problem (general optimisation, or linear programming) adequately defines just which simplex method is in use.

Algorithms using the simplex method start with a feasible point, and then generate a simplex either by random shifts or defined steps along each of the dimensions. Given this initial simplex, the worst point (i.e. for maximisation problems, the minimum function value) is discarded and the coordinates reflected through the centroid of the simplex, arriving in an area which is usually more optimal. The original formulation (Spendley et al. 1962) dealt with a regular simplex, with manual adjustment required in the near-optimal region to prevent over-shooting. Subsequently, the refinement of Nelder and Mead (1965) (incorporating contraction and expansion steps during each of the iterations) has resulted in a robust general algorithm, capable of searching across the hyperspace before shrinking onto an optimum. The usual termination criteria is when all points, having contracted into an optimum, display approximately equal function values. The required precision of the termination criteria can be specified by the user. As with other techniques, scaling of the variables is recommended,

although this algorithm can expand along any of the 'longer' dimensions to accommodate moderate departures in scale.

This method has been observed to flatten onto a constraining plane, when the optimal region lies along this plane. This effective loss of one dimension in the simplex can cause problems when another constraint becomes active and the method needs to 'turn the corner' (Box 1965). To cater for this, the 'complex' was proposed (Box 1965), which is effectively a simplex with more than (n + 1) vertices. The iteration technique remains the same. Box (1965) suggested 2n vertices, but concluded that this choice is not critical. Whilst intuitively the complex should perform better than the simplex, in practice this does not appear to be the case (Box 1965; J.A. Nelder, pers. comm. 1992), and this method has not been widely used.

The simplex algorithm has been proven convergent, on strictly convex functions in lower dimensions (Lagarias et al. 1998). However, more generally its theoretical properties have not been sufficiently investigated (Polyak 1987). In empirical test studies on smooth functions with few dimensions, the simplex method is clearly inferior to the gradient methods (Fletcher 1987, Polyak 1987). However, otherwise it has proven itself to be a very robust method across a wide range of real-world problems (Barabino et al. 1980, Bunday 1984). It remains a cornerstone in some of the well-known software libraries, such as the Numerical Algorithms Group (NAG) and the International Mathematical and Statistical Library (IMSL).

CONSTRAINED OPTIMISATION

Constraints cannot be directly incorporated into the search method, leaving the penalty function or feasible directions solutions as possibilities. Using a penalty or barrier function of adding some large value (as previously outlined), any projection outside the feasible region would be rejected. A smaller (contraction) step would then be used to obtain a new point, presumably within the feasible region and close to the constraint. If non-feasible points are projected back onto the constraint (the method of feasible directions), each new point in this direction will be on this optimal plane, unless the constraint is still a slight 'over-shoot' of the optimal value, in which case further iterations will contract away from the constraint. In practice, either of these constraint methods should work reasonably well with direct search techniques.

PROBLEMS

In severely constrained situations, the initial generation of the required number of vertices can be of concern, but generally this is not a problem. As with other methods, possible convergence onto a local optimum remains the major problem. Once all the vertices of the simplex are within the basin of attraction of a local optimum, this method will converge to it. The simplex method has been shown to search around the Euclidean space more thoroughly than the gradient method (Mayer et al. 1991), resulting in a higher proportion of starting points converging to the global optimum. However, convergence to a local optimum remains a problem that only multiple starting-points or some random perturbation scheme can attempt to address.

AGRICULTURAL MODEL APPLICATIONS

Despite their theoretical shortcomings, direct-search methods have proven effective in a number of studies. For the calibration of grain-crop models to field data, both the simplex direct-search algorithm, and its extension to the complex, have been used successfully (Hammer and White 1992, Hammer et al. 1993, Olsen et al. 1993). Hendrickson et al. (1988) showed the Hooke and Jeeves pattern-search algorithm to be a robust method, and preferable to the gradient method, for the calibration of a rainfall-runoff model. On a similar problem, an enhanced version of the complex algorithm proved superior to a genetic algorithm (Franchini et al. 1998).

The simplex algorithm has also performed moderately well in the optimisation of the economic performance of agricultural models. Mayer et al. (1989, 1996a) demonstrated its superiority to gradient methods on a dairy farm model. Botes et al. (1996) found the simplex to be more flexible and realistic than dynamic programming on a crop irrigation problem, and Parsons (1998) showed it to have (at best) comparable performance to a genetic algorithm in determining an optimal silage harvesting plan. Conversely, both Hart et al. (1998) and Kuo et al. (2000) showed the simplex to be inferior to a genetic algorithm, in the optimisation of dairy farm and irrigation scheduling models, respectively.

4. SIMULATED ANNEALING

Simulated annealing is a term covering an evolving family of algorithms based on thermodynamical models (Davis and Steenstrup 1987). In the past, these methods have variously been called stochastic cooling, Monte Carlo annealing, probabilistic annealing, stochastic relaxation and probabilistic hill climbing. Their common link is that, by probabilistically accepting 'backwards' (or less successful) steps, they can generally escape from local optima, to eventually find the global optimum. Their main disadvantage is the often excessive computation required to statistically guarantee this (van Laarhoven and Aarts 1987).

Simulated annealing works by mimicking the annealing (cooling) process of metallurgy. This occurs when a molten substance is cooled slowly, allowing the particles to gradually align to reach a near-minimal energy state of mutual forces (Bohachevsky et al. 1986). If cooled rapidly, or quenched, many local energy maxima will be formed in the solidified product, and the highly-ordered, crystalline state of lowest energy will not be found (Corana et al. 1987).

An algorithm to simulate cooling to the thermal equilibrium, based on the Boltzmann distribution, was originally proposed in Metropolis et al. (1953). Its application to optimisation problems was independently developed by Kirkpatrick et al. (1983) and Cerny (1985). In this algorithm, new points are sequentially generated, usually at random from a nominated probability distribution. If the new point has a lower energy content (in a minimisation sense), it is automatically accepted. If not, i.e., if the new point is less successful, it is subjected to the Metropolis criteria –

$$P = \exp (- E / k_B T), \text{ where}$$

P is the probability of acceptance,
E is the change in energy values between the two points,
k_B is a parameter controlling the temperature schedule, and
T is the temperature at this stage.

As with true annealing, the results are largely controlled by the temperature process. At higher temperatures, 'backwards' steps are more likely, although not if they are greatly backward. At lower temperatures, nearer the solution, fewer backwards steps are accepted (Radcliffe and Wilson 1990). This behaviour allows the algorithm to initially search widely around the feasible hyperspace, and then zero in on the optimum as the temperature approaches the 'freezing point'. It is important to spend a considerable amount of effort in this stage, so that local instabilities (or sub-optimal arrangements) can gradually be removed (Kirkpatrick et al. 1983).

Hence the rate of temperature decline throughout the operation, termed the temperature schedule, is of critical importance.

It has been proven mathematically that the probability of finding the global optimum approaches one, as the number of transitions (or trial values) tends to infinity (van Laarhoven and Aarts 1987). Herein lies an obvious problem, in that infinite runs are never possible. Under traditional Boltzmann annealing, the temperature declines inversely logarithmically with time, ensuring a slow and thorough search. However, even in this situation there is no absolute guarantee, and the method may still fail to find the global optimum of a multiple-optima system (Ingber 1993).

The efficiency of Boltzmann annealing was improved with the development of fast annealing by Szu and Hartley (1987). They proposed replacing the Gaussian sampling distribution with the Cauchy, with the fatter tail of the latter allowing more efficient sampling (Ingber and Rosen 1992). Also, fast annealing uses a more rapid rate of temperature decline, namely inversely linear with time, giving a method which is exponentially faster than the original.

Ingber (1989) introduced a further series of improvements, to arrive at a method termed 'very fast simulated reannealing'. Firstly, the multi-dimensional Cauchy distribution is used, rather than applying the easier one-dimensional form across the required number of dimensions. Secondly, the temperature decreases inversely exponentially with time, to give a more rapid search. Thirdly, reannealing introduces the process of adjusting the search in each dimension by adapting to the changing sensitivities of each parameter. This is implemented regularly during the search. In practical terms, reannealing results in the use of smaller steps with the more critical parameters, but larger searches along the insensitive dimensions.

'Adaptive simulated annealing' was introduced in Ingber (1993). This method incorporated all the above advances, and added the refinement of dynamically adjusting the sensitivities of the search in each of the dimensions (hence the 'adaptive' feature appears a refinement of reannealing).

Recent trends are towards 'more greedy' or faster implementations, termed simulated quenching or tempering. These use a more rapid temperature decline, usually applied as a scaling factor on the rate of temperature decrease. These methods are computationally quicker, but more risky, as they are more likely to converge to a sub-optimal solution. However, this may be the only practical method of using the advantages of simulated annealing on very large problems, as conservative temperature schedules could take almost infinite time. Thus far, simulated quenching has been shown to be fairly robust, although only over a limited number of

applications (Ingber 1993). Optimal control parameters of these methods for a range of systems are yet to be determined.

APPLICATIONS

As may be expected with a robust optimisation technique, applications of simulated annealing and comparisons with other optimisation methods are becoming more common. Evaluations of simulated annealing on a suite of mathematical test functions (mostly of low dimensionality) are listed in Bohachevsky et al. (1986), Corana et al. (1987), Styblinski and Tang (1990), and Ingber (1993), with these generally confirming it as a successful optimisation method. Kirkpatrick et al. (1983) demonstrated that simulated annealing was surprisingly good at solving examples from the difficult class of NP-complete (nondeterministic, polynomial-time complete) problems, such as the travelling salesman problem, and computer design and component placement. Other allocation-type problems which have also been solved by simulated annealing include design, routing, and image processing problems (van Laarhoven and Aarts 1987), the allocation of the protein structure in squash seeds (Holak et al. 1989), and the visual determination of geometric shapes using very fast simulated reannealing (Wu and Levine 1994). Statistical applications have also featured prominently, including evolutionary tree design (Lundy 1985), the determination of optimal experimental designs (Bohachevsky et al. 1986, Morris et al. 1993), the estimation of maximum likelihood parameters (Goffe et al. 1994), and the solution of over-parameterised nonlinear regression models (Mayer et al. 1996b). A difficult optimisation of groundwater remediation strategies via modelling was solved successfully by simulated annealing (Kuo et al. 1992). This problem had previously defeated hill-climbing methods, which tended to converge to sub-optimal solutions.

A number of applications to agricultural systems have also been successful. Considering the harvesting schedule for forestry blocks, Lockwood and Moore (1993) list a range of simulated annealing applications, with one covering some 27 500 blocks of trees. This is far in excess of the limit of 200 suggested by Roise (1990) using the conventional method of conjugate gradients. Bos (1993) also used simulated annealing to solve a forestry management problem, which had proved too computationally intensive for the branch and bound method. On a temporal harvesting model of a prawn fishery, Watson and Sumner (1999) adopted simulated annealing to avoid being trapped by the many local optima. The CERES-Maize agricultural model was calibrated to USA data (Paz et al. 1999), using the simulated annealing implementation of Goffe et al. (1994).

5. TABU SEARCH STRATEGY

The basic ideas of tabu (or taboo, Cvijovic and Klinowski 1995) search were developed in the field of combinatorial optimisation, over a number of years (Glover 1989). In this optimisation method, locations (being defined as vectors of the independent variables) which have previously been visited are earmarked as 'tabu', and thus not allowed to be re-visited. Under this constraint, the search scheme is forced away from current locations, which is a desirable feature if these optima are only local. By continually rejecting these sub-optimal regions, the tabu search method should eventually find the global optimum. Independently, Hansen and Jaumard (1990) developed their 'steepest ascent mildest descent' method in a study on the maximum satisfiability problem, first proposed in a conference in 1986. In practical terms this strategy is identical to tabu search, and the latter terminology has subsequently been adopted in the literature.

The tabu search method is controlled by a number of operational parameters and strategies (Glover 1990a), the most important of which appears to be the length of, and method of maintaining, the tabu list. The reactive tabu search (Battiti and Tecchiolli 1994a) is a dynamic application which incorporates a number of the more advanced strategies, but these appear to be heuristics which are limited to their combinatorial optimisation problem. Tabu search is also a metastrategy which can and has been used with a number of other optimisation methods. In practical usage, it has predominantly been combined with hill-climbing techniques or problem-specific heuristics.

Tabu search has been applied to a wide range of practical problems, including scheduling, allocation, sequencing, patterning, and planning applications (Glover 1990b). These problems are generally discrete in nature, but tabu search can be adapted to continuous problems by discretising each dimension (Cvijovic and Klinowski 1995), at the desired level of precision. This is similar to the approach used in binary genetic algorithms, and again mapping using Grey coding is recommended (Battiti and Tecchiolli 1994a).

On combinatorial optimisation types of problems, tabu search has been shown to be most efficient, and it has out-performed a number of problem-specific heuristics (Glover 1989, Hansen and Jaumard 1990). On smaller or simpler problems, it generally performs similarly to simulated annealing (Battiti and Tecchiolli 1994b, Cvijovic and Klinowski 1995). However, with moderately-sized or large problems tabu search has been shown to be superior to simulated annealing, in terms of both solution quality and speed of convergence (Hansen and Jaumard 1990, Osman 1993, Battiti and Tecchiolli 1994b). On these larger applications, however, tabu search can

have problems maintaining an adequate length of the memorised tabu list (Glover 1990b). In the initial studies, only small tabu lists (of the order of 5 to 12) needed to be maintained (Glover 1989). With larger problems, much larger lists are needed to prevent cycling back to sub-optimal solutions (Glover et al. 1993). Alternatively, Battiti and Tecchiolli (1994a) incorporate an escape heuristic to cope with problems where the search scheme is captured by wide local optima.

This is indicative of the main problem concerning the wider application of tabu search to multi-dimensional studies. The majority of tabu applications have been combinatorial optimisations, which are mostly of a spatial (or temporal) allocation nature. Here, the search is basically in two dimensions, with the size of the problem being dependant on the number of nodes, objects and/or pathways to be allocated or assigned. Taking for example the quadratic assignment problem with L locations, the size of this problem is of order L^2 (Battiti and Tecchiolli 1994a). If larger studies of this type are causing problems with tabu list lengths, then there appears little hope for true multi-dimensional applications which are of the order X^d, where d is the dimensionality and X the number of divisions in each dimension (and thus dependent on the required level of precision).

This problem of the tabu search strategy is intuitively easy to understand. On arriving at a local optimum, the method begins searching around the local neighbourhood. Whether this is via complete evaluation or some neighbourhood subsampling scheme (Cvijovic and Klinowski 1995) is irrelevant here. Each surrounding point must be investigated and added to the tabu list to force the search away, into (hopefully) more profitable zones. Unless the entire surrounding region is excluded via assigning all points within it to tabu status, the method will keep sliding into adjacent dimensions to find 'near-optimal' solutions, rather than accepting the seemingly sub-optimal values. Hence, it will eventually return to the local optimum unless prohibitively long tabu lists are maintained. For example, a farm's profitability may primarily depend on the interaction between the key management variables of stocking rate, pasture species, irrigation, and fertiliser application. If a locally-optimal combination of these has been found, all surrounding locations (across all possible dimensions) will need to be added to the tabu list. However, the 'more minor' (in terms of their effect on farm profitability) management decisions, such as bull selection and ratio, mating period, culling rates and times, weaning, grazing patterns, etc., allow ample opportunity for the search to 'slide sideways' into these adjacent dimensions, whilst staying close to the local optimum of the key dimensions.

Mathematically, consider a local maximum in a problem of d dimensions. If the 'valley' (being the barrier to a higher, more optimal region) lies n nodes (values of the independent variables) away, then N, the number of

neighbourhood points to be investigated and retained in the tabu list, is given (Mayer et al. 1998b) by –

$$N = (2n + 1)^d - 1$$

The usual 'curse of dimensionality' applies here. Even with a moderately-sized maximisation problem of 10 dimensions, this gives N of about 10 million for a dividing valley only 2 nodes away, or 3 500 million for 4 nodes. Maintaining and checking tabu lists of these lengths is clearly infeasible. Obviously, no real-world problem will be symmetrical and equally-responsive in each of the independent variables being optimised, so the above equation is a guide only. The distance to the nearest ridge will probably be a different number of nodes away in each dimension, and some dimensions may well respond in a long, almost-flat slope or plateau. Regardless, this analysis gives an indication of the excessive length of tabu lists which would need to be maintained for highly-dimension or high-precision problems.

APPLICATIONS

Tabu search has been used widely on a range of allocation-type problems, and test functions which have only lower dimensionality (Battiti and Tecchiolli 1994a). Cvijovic and Klinowski (1995) used it on continuous test functions of up to 6 dimensions, but limited the number of divisions (nodes) in each of the dimensions, relying on a hill-climbing algorithm to finalise the optimisations. Faced with the difficulties (as outlined above) of even 6 dimensions, they only considered a crude level of discretisation in each dimension, to reduce the problem to a manageable size. This approach is unlikely to be useful in a practical sense with a large number of truly continuous variables.

Tabu search has yet to be applied successfully to the optimisation of agricultural systems models. Hart et al. (1998) adopted a tabu-like list with a genetic algorithm in his study with a New Zealand dairy farm model. Here, previously-visited locations were stored in the computer's memory so that they did not have to be re-evaluated if called upon again. Mayer et al. (1998b) also used this facility, in combination with a simulated annealing optimisation of a dairy farm model, and struck problems regarding the necessary level of resolution of the continuous variables in the final stages of the optimisation. However, these approaches cannot be considered true applications of the tabu strategy, as the stored locations are merely used to save on computer time (by not having to re-run the simulation model). The

resultant values of the optimisable variables (farm milkfat production, and profit, respectively) are still returned and used in the respective optimisations, rather than this trial combination of input variables being excluded from consideration, as would be required under a tabu strategy.

6. HYBRID METHODS

Whilst many different variants exist of most of the basic optimisation techniques, they do tend to have their own distinctive respective methodologies. For example, hill-climbing, tabu search and simulated annealing each have one current solution (with perhaps some memory facility of the previous best, especially after a 'less-successful' move), whilst the simplex method and evolutionary algorithms maintain populations of solutions. Similarly, each method has its own particular rules regarding the generation and selection of new candidate solutions, as previously outlined.

Hybrid methods involve the combined use of two or more of these basic techniques, either sequentially or in parallel. The most common hybrids in the literature appear to be genetic algorithms followed by an alternative method to 'fine-tune' the solution. The premise used here is that the genetic algorithm (usually binary) is good at the exploration phases of the search, and this is continued until most population members have converged to the near-optimal region. At this time, the discrete nature of the coding allegedly interferes with final convergence, so the hybrid method then switches to a local optimiser, such as the simplex (Wang 1991), or one of the gradient methods (Franchini 1996, Mayer et al. 1996a, Campbell et al. 1998, Franchini et al. 1998, Hart et al. 1998, Goggos and King 2000). Using two optimisation methods in sequence obviously involves more user effort, including extra computer coding, and decision rules on when best to switch across to the alternate algorithm. Most of these published studies claim this hybridisation to be a success, but (where listed) the degree of improvement over the genetic algorithm solution appears small – 0.13% in Mayer et al. (1996a) and 0.5% in Hart et al. (1998). Whether or not this improvement could equally have been gained by finer precision of the binary codings, better tuned options, and/or longer runtime, remains open to conjecture. Campbell et al. (1998) progressed from a hybrid of a binary genetic algorithm with a gradient method, to a real-valued genetic algorithm. This approach may well provide the best solution for systems where fine-tuned convergence is required.

As opposed to running two or more algorithms sequentially, the true integration of different optimisation methods has proven to be more difficult. Annevelink (1992) incorporated a problem-specific local optimiser into the

fitness evaluation phase of a genetic algorithm, yet overall this remained primarily an implementation of an evolutionary algorithm. Because of their very nature, incorporating evolutionary algorithms with other optimisation algorithms - even the generic tabu search (Ibarki 1997b) appears difficult. Ibarki (1997b) outlines the potential use of a simulated-annealing-like probabilistic acceptance scheme in the selection of parents or new population members, and Varanelli and Cohoon (1995) incorporated a simulated annealing heuristic on a combinatorial optimisation problem. However, neither these nor other adaptations have appeared subsequently in the applied literature.

Conversely, the tabu search strategy is well suited to true integration with other schemes, in particular simulated annealing (Fox 1993, Osman 1993, Ibarki 1997b) and greedy randomised search (Feo and Resende 1995). However, these studies are all combinatorial optimisation applications, where the specific allocation-style nature of the tabu search logically aligns with the problem. Also, despite the claims, many of the hybrids listed in the literature remain of one basic optimisation type, with only one or more heuristics adopted from the alternate optimisation method. Just which heuristics are worthwhile appears to depend on the problem type.

As yet, there is no overall agreement as to which of the hybrids work best, especially regarding the optimisation of systems models. Their best use would be in the adoption of useful heuristics from one method into another, but the profitable use of these cross-overs appear very problem-specific.

Chapter 5

COMPARISONS OF OPTIMISATION TECHNIQUES

In this chapter, theoretical studies which variously prove global convergence, or compare rates of convergence, are listed and contrasted for a range of optimisation methods. These results then lead to a consideration of empirical studies, on test functions, combinatorial optimisation problems, and systems models. Within the latter two classes, applications of different optimisation methods to agricultural systems are compared and discussed. It is concluded that when considering the optimisation of systems models across the range of problem types, evolutionary algorithms are likely to be at least as good as, and probably superior to, the other available optimisation methods.

1. INTRODUCTION

It is interesting to note that, in a number of disciplines, most optimisation methods have been proven convergent to the global optimum of the system, as the number of iterations approaches infinity. Using Markov chain theory on combinatorial optimisation problems, this property holds for simulated annealing (van Laarhoven and Aarts 1987), genetic algorithms which incorporate elitism (Peck and Dhawan 1995), evolution strategies (Bäck and Schwefel 1993), and evolutionary programming (Spears 2000). These results are consolidated in (Fogel 1995b), which shows that all of the different variations of evolutionary algorithms are globally convergent, provided they incorporate elitism (which means the $(\mu+\lambda)$ scheme of evolution strategies). Regarding rates of convergence on combinatorial optimisation problems, the expected quality of the simulated annealing

solution is less than or equal to that of evolutionary algorithms (regardless of binary or real-valued coding, with or without mutation, and with or without recombination, Hart 1996), and also less than or equal to repeated local random search (Ferreira and Zerovnik 1993). However, these results are more of academic interest only. They cover only a subset of potential optimisation problems, and are thus not directly applicable to large, general modelling studies. Also, the basic assumption of infinite iterations is unrealistic - if these were truly possible, the optimum would be easily found by complete enumeration. What is needed in practice are methods which give the best chance of finding the optimum (or a value acceptably close to it), within a reasonable or (at worst) feasible time frame.

No discussion of optimisation theory would be complete without consideration of the 'no free lunch' theorem (Wolpert and Macready 1997). The basic premise here is that across all possible problem types, the average performance of all optimisation algorithms is equal, which further implies that the past performance of any algorithm has no possible bearing on its performance on future problems (Wolpert and Macready 1997). These ideas have formed the basis of considerable discussion and debate, particularly on the Genetic Algorithms Digest (http://www.aic.nrl.navy.mil/galist/). Whilst theorists maintain this theorem to be mathematically proven and thus true, practitioners have tended to take a more pragmatic view – 'across all possible problem types' is an unrealistic concept, as problem types tend to divide into a number of discrete categories. Within each of these categories, previous studies on similar problems can surely be used as a guide to the expectation of future performances.

Given the wide range of both optimisation techniques and the types of problems that they can be applied to, comparisons between these can be difficult to overview and interpret. An addition complication is that in many cases the developers or supporters of a particular algorithm have published empirical results showing their method to be superior to other contenders. It must be appreciated that these researchers are naturally up-to-date and will tend to have well-tuned parameterisations of their favourite method, and may well be testing this against out-of-date or ill-parameterised versions of their competitors. Also, perhaps subconsciously, they may be using test cases upon which they know or suspect that their algorithm will work well.

As is intuitively expected, and proven by Wolpert and Macready (1997), there is no 'globally optimal' method of optimisation. Each technique has features which enhance its performance on particular types of problems, and alternately give the expectation of poorer performance on other problems. As outlined in Chapter 2, agricultural systems generally posses a range of properties which make optimisation difficult, and we thus need to look for methods which will perform well under these conditions. In other

disciplines, the more recently-developed algorithms have been rapidly gaining acceptance, at the expense of the more traditional hill-climbing and direct-search methods. For example, of the 21 papers in the 'Metaheuristics in Combinatorial Optimization' issue of 'Annals of Operations Research' (Volume 63, 1996), eight used tabu search, six applied simulated annealing, and five used genetic algorithms.

2. TEST FUNCTIONS

Mathematical test functions have traditionally formed the basis for comparisons between the different optimisation methods, and have been widely reported in conference proceedings and journals. These formulated functions range markedly in form and difficulty – from single-variable up to highly multidimensional; functions with a single optimum to those with multiple (local) optima; smooth surfaces with wide basins of attraction to more deceptive functions with quite narrow optima; and those with independence between the dimensions versus functions with interacting (for example, multiplicative) variables. As may well be expected, the performance of the available optimisation algorithms varies across these different classes of test functions. Also, results do appear to be correlated with the optimisation method backgrounds of these authors.

On Rosenbrock functions of up to 10 dimensions, Corana et al. (1987) showed simulated annealing to be superior to the simplex direct-search method. Other supporters of simulated annealing, Ingber and Rosen (1992), showed it to perform better than or equal to a genetic algorithm, on functions of up to five dimensions. However, on their 30-dimensional function, the genetic algorithm repeatedly identified better optima than simulated annealing, despite taking longer to achieve these results. Mayer et al. (1991) showed that the simplex method outperformed hill-climbing, whereas Goffe et al. (1994) found both hill climbing and the simplex to be far less successful at escaping local optima than simulated annealing, which converged to the global optimum every run. Against these results, Battiti and Tecchiolli (1994b) demonstrated the tabu search strategy to be superior to simulated annealing, and on low-dimensional functions Cvijovic and Klinowski (1995) found tabu to be markedly more efficient than simulated annealing, hill-climbing and random search. Keane (1996) showed various evolutionary algorithms to be superior to simulated annealing, and within the former genetic algorithms identified better optima than either evolutionary programming or evolution strategies. This contrasts with Bäck and Schwefel (1993), where evolution strategies gave better results than genetic algorithms and evolutionary programming, and Fogel (1995a), in which evolutionary

programming beat genetic algorithms. In a more comprehensive comparison across eight test functions (Fogel 1995b), evolutionary programming was superior on five, and genetic algorithms on two, with one tie. Hinterding et al. (1995) also showed varying performance of the different variants of evolutionary algorithms – genetic algorithms proved superior on some test functions, and evolution strategies on others.

One result contrary to the expectation of better performance from the more modern methods is listed in Styblinski and Tang (1990), where a stochastic version of the conjugate-gradient (hill-climbing) method outperformed simulated annealing. The test functions used here were polynomial-type with minor local perturbations, roughly conforming to the general smooth, unimodal shape where hill-climbing algorithms should succeed. It is unlikely that this result would carry over to more irregular surfaces.

The major problem with basing strong inferences on test function results, however, lies in their very basis. Despite the possible range of complexities, they remain only test functions – as such, they are mainly smooth, of a similar scale (in each of the different tested dimensions), and frequently do not involve interactions between the dimensions. As demonstrated previously, this is most unlike the expected response surfaces of agricultural systems models. As Saloman (1996) and Hammel (1997) contend, these results are unlikely to extend to the optimisation of real-world problems.

3. COMBINATORIAL OPTIMISATION PROBLEMS

As with test functions, a multitude of comparative studies have been and are appearing in the scientific literature. Combinatorial optimisations are basically allocation or scheduling problems, and are essentially Euclidean in nature (i.e., they can be laid out in two dimensions, with varying numbers of nodes, pathways or objects to allocate). They are thus true real-world problems, and despite being very computationally challenging as the number of objects increases, they only represent the 'lower-end' of problem dimensionality.

Studies which reported the values of the optima identified by the compared methods are summarised in Table 1, which does show somewhat mixed results. Here, both simulated annealing and tabu search appear to have the advantage, consistently returning optimal or near-optimal values. This relativity is maintained in Feo and Resende (1995), where these two were superior to a genetic algorithm. The best method in this study was their implementation of a greedy randomised adaptive search, with this algorithm also performing quite well in Ibarki (1997a). Genetic algorithms, including

hybrids, occasionally performed best in combinatorial optimisations (Table 1), but do not appear overly well-suited to these. Between the different forms of evolutionary algorithms, genetic algorithms perform best on some problems, and evolution strategies on others, as was found in laminate design (Le Riche et al. 1995) and networking problems (Rothlauf et al. 2000).

Table 1. Relative performance (average % from global optimum) of optimisation methods[#], on combinatorial optimisation problems. Bold entries indicate the best method(s) within each study.

Source	Random	HC	Heur.	Tabu search	SA	GA	Hybrid (type)
Hesser et al. (1989)			**0.00**		**0.00**	**0.00**	
Osman (1993)			26.7	**0.37**	1.29		
Park (1995)					**0.78**	1.17	
Glass and Potts (1996)		0.58		1.11	0.36	1.51	**0.28** (GA/HC)
Houck et al. (1996)	2.21		5.75			**0.29**	
Ibarki (1997a)	0.05			0.09	**0.01**		
Ibarki (1997b)				0.09	**0.02**	0.53	

[#] Random = Random methods, HC = Hill-climbing (gradient), Heur. = Heuristic (problem-specific), SA = Simulated annealing, GA = Genetic algorithm (binary)

A number of other studies on allocation-type problems (Glover 1989, Hansen and Jaumard 1990, Osman 1993, Battiti and Tecchiolli 1994b) show examples where tabu search is superior to simulated annealing (in either terms of the optimal value obtained, speed of convergence, or both). Overall, tabu search would thus appear the 'best bet' for these types of problems, not withstanding possible authors' bias and the 'no free lunch' theorem (Wolpert and Macready 1997). This is probably because its methodology is directly related to optimisations of a Euclidean nature. However, it is unlikely that this result will extend to the optimisation of more general systems.

4. SYSTEMS MODELS

The comparison of different optimisation methods on systems models is of direct relevant to this monograph. Again, a range of studies has been published, as summarised in Table 2 for publications which have listed or graphed the respective optima of the compared algorithms. This list is far from complete, as the use of modern optimisation algorithms across a wide

range of application areas has snowballed in the past few years. However, we believe it to be fairly comprehensive in its coverage of agricultural systems – these types of applications dominate this table, particularly in the more recent publications.

Table 2. Relative performance (average % from global optimum) of optimisation methods[#], on the optimisation of simulation models. Bold entries indicate the best method(s) within each study.

Source	HC	Sx	Heur.	SA	GA_b	GA_r	ES	Hybrid (type)
Bramlette and Cusic (1989)	0.95			**0.00**	0.06			
Mayer (1989)	11.1	**3.00**						
Bramlette and Bouchard (1991)	0.95			**0.00**	0.06			**0.00** (GA_b/HC)
Karr (1991)		**0.00**		**0.00**				**0.00** (GA_b/Sx)
Wang (1991)					0.08			**0.00** (GA_r/Sx)
Franchini (1996)					5.60			**0.00** (GA_b/Heur.)
Mayer et al. (1996a)	15.3	5.7		**0.00**	0.32			0.19 (GA_b/HC)
Hart et al. (1998)	5.93	21.1			2.02			**1.47** (GA_b/HC)
Parsons (1998)		5.20			**0.00**			
Verryn and Roux (1998)			14.7		**0.00**			
Mariano (1998)			11.3		**0.00**			
Kozan (1999)			7.9		**0.00**			
Mayer et al. (1999a)				1.02	**0.12**			
Mayer et al. (1999b, 2001)					**0.04**	0.05	0.26	
Meszaros et al. (1999)			0.13		**0.00**			
Pabico et al. (1999)	2.02				**0.00**			
Kuo et al. (2000)	7.36			2.05	**0.00**			
Lu and Eriksson (2000)			7.13		**0.00**			

[#] HC = Hill-climbing (gradient), Sx = Simplex (direct search method), Heur. = Heuristic (problem-specific), SA = Simulated annealing, GA_b = Genetic algorithm (binary), GA_r = Genetic algorithm (real-value), ES = Evolution strategy

Throughout the earlier days of computational optimisation, only the more traditional hill climbing and direct search methods were available, and these have been used to effect on occasions. Despite the perceived theoretical advantages of the former, it is evident that the more robust simplex algorithm performed better on real-world models. This is probably largely

due to the irregularities and complexities of these problems, resulting in a non-smooth response surface, for which gradient methods are not well suited. Similarly to these results, Hendrickson et al. (1988) showed that the Hooke and Jeeves direct-search method outperformed a number of hill-climbing methods, on the calibration of model parameters with a 16-dimensional rainfall/runoff model. Whilst hill-climbing methods show rapid convergence once within the basin of attraction around the global optimum, this feature also means they are prone to convergence to local optima (Horton 1996, Mayer et al. 1996a, Hart et al. 1998), and thus are not well suited to the optimisation of models.

Tabu search is conspicuous by its absence in Table 2. Despite theoretical and demonstrated advantages in combinatorial optimisation type problems, it has yet to be used successfully on agricultural systems models. The underlying methodology of tabu search may be incorporated with other optimisation methods (Fox 1993, Osman 1993). This approach offers obvious potential for future problems, particularly if they contain spatial or temporal allocations.

In the more recent years, the newer developments of evolutionary algorithms and simulated annealing have clearly shown their value (Table 2). These methods have consistently identified better optima than the earlier techniques, although usually at the expense of longer search times. The genetic algorithm of Meszaros et al. (1999) proved most efficient, identifying a superior optimum in just 10^4 model runs whilst a grid-search heuristic took an average of 10^7. A number of initial studies (Table 2) used hybrid methods with some success. However, as discussed previously, their improvement in realistic terms over the basic method (usually an evolutionary algorithm) is not great, especially considering the extra programming required and that the additional precision in the optimum could probably equally be obtained by a finer-tuning of the coded options, or running the evolutionary algorithms longer.

The thorough search pattern of simulated annealing usually finds the global optimum, but this degree of success is related to the difficulty (in particular, dimensionality) of the problem. Table 2 shows a 'perfect score' for Bramlette and Cusic (1989), Bramlette and Bouchard (1991), Karr (1991), and Mayer et al. (1996a), with these studies having dimensionalities of 8, 8, 10 and 16 respectively. At these levels, simulated annealing can still cope, although the 16-dimensional study took 10^5 model runs to converge. In the two studies which reported sub-optimal performance of simulated annealing, Kuo et al. (2000) had 14 dimensions and Mayer et al. (1999a) had 20 and 40 independent options. Interestingly, Ingber (1996) states that simulated annealing is computationally limited to problems of about 15 to 20 dimensions, and recommends simulated quenching for larger studies.

However, as previously noted this method is not foolproof, and can converge to local optima (Ingber 1993, Mayer et al. 2001).

The more recent studies in Table 2 point to the superior performance of evolutionary algorithms on these modelling problems. Binary genetic algorithms have been used most often, although real-value evolutionary algorithms have appeared of late. Here, we must admit to potential author bias – as our past publications indicate, binary genetic algorithms have been our preferred method of optimisation, and hence the most targeted in literature searches. However, other practitioners in the agricultural systems field appear to have adopted a similar path, also with success. It is unlikely that this trend would be greatly different for other disciplines.

Chapter 6

ROBUST PARAMETERS FOR EVOLUTIONARY ALGORITHMS

Whilst evolutionary algorithms have been well tuned for a number of different problems, finding the best combination of operational types and parameters remains an obvious challenge. In this chapter, various methodologies from the different underlying types of evolutionary algorithms are compared and contrasted in turn, particularly regarding the contribution of each option to the key processes of exploitation and exploration. The degree to which these options interact is then discussed, and an overall combination of 'best bet' operators is suggested. In the absence of other problem-specific information, this is recommended as a robust combination for the optimisation of future agricultural systems.

As well as varying in their basic philosophies (i.e., genetic algorithms, evolution strategies or evolutionary programming, as overviewed in Chapter 3), each of the forms of evolutionary algorithms has a wide range of operation types and parameters. The overall combination of all these operators controls the crucial balance between exploitation (using and combining the existing genetic material in the population to best effect) and exploration (searching for better genes). Overweighting of exploitation can lead to premature convergence (to a local optimum), as the algorithm focuses in on the area around the current best solutions. Conversely, over-exploration can waste time by continually breaking away and re-looking at sub-optimal areas of the search-space.

One valid criticism of evolutionary algorithms is that, unlike most of the competing optimisation methods, they have no definite termination criteria. Pelikan et al. (2000) and others have shown that the number of generations required for convergence is proportional to the square root of the problem size, and inversely proportional to selection pressure. Here, these

considerations also interact with population size – for a given amount of computing resources, is it better to have a smaller population and more generations, or vise-versa? Even when all (or most) of the population members have converged to an 'apparently optimal' set of genes, the mutation operator continually seeks to break solutions away from this, searching for more-optimal regions. In practice, most users of evolutionary algorithms tend to run them until 'apparent convergence', i.e., until the value of the optimum has not changed (or changed only minimally) over a large number of generations. As with all optimisation methods, replicated runs which converge to the same solution increase the user's confidence that this is the global optimum.

Finding the best combination of operational parameters for any given evolutionary algorithm and problem is extremely difficult, if not impossible (Michalewicz and Fogel 2000). Given the number and options of each parameter, a complete evaluation will never be feasible. Davis (1991b) and others have suggested using a 'higher-level' genetic algorithm to optimise the parameters of the 'lower' genetic algorithm. Unless trialed on only a very small or simple problem, this would also require too many runs to be practical, as each trial member of the population of the higher-level genetic algorithm requires a complete optimisation of the lower-level genetic algorithm. This effectively leaves any fine-tuning to be done 'manually', by the user. Here, each parameter is usually investigated in turn, whilst holding all others at 'near-optimal' or at least robust values. However, even this approach falls by the wayside when considering the often interacting effects of these operational parameters (Michalewicz and Fogel 2000). As we have found (Mayer et al. 1997, 1999a, 2001), in practice finding even near-optimal parameter settings can take considerable trial and error, with months of computational effort required to tease out these interactions – and even then, any 'definitive' results still tend to be partially confounded with each other.

The different operational parameters of evolutionary algorithms have varying degrees of effectiveness – depending on the problem, some operators are critical to efficiency, whilst others may not be so important. As the key operators of recombination and mutation have been shown to have a synergistic effect (Fogel 1995b), the chosen evolutionary algorithm should use both. Other key decisions include the best coding of the problem under consideration, selection methods for both the parents and offspring, and the size and management of the population.

CODING OF MODELLED OPTIONS TO GENETIC REPRESENTATION

The first choice facing the practitioner is the appropriate coding mechanism. The two commonly-used options are binary coding (where each of the model's management options is mapped onto one or a number of binary genes, with this number being dependent on the precision required for each), and real-value coding (where each gene is a numerical representation of one of the options being optimised). Of course, any level of alphabetic representation may be used profitably - Moore et al. (2000) used a three-level variable in a trinary genetic algorithm, because their forest harvest scheduling problem had three natural options.

Supporters of the different evolutionary algorithm 'camps' have each reported the superiority of 'their' method. Pabico et al. (1999) and others claim that binary coding is preferable, even for the representation of continuous variables, because this makes best use of schemata. Conversely, Michalewicz (1996), and Fogel and Angeline (1997) suggest that the coding should represent the problem at hand, and that the one-to-one option-to-gene correspondence has a more 'natural' feel and performance.

In practice, neither argument stands up for all problems. Typically, an agricultural model will have a mixture of discrete options (for example, which species of pasture to plant, whether or not to apply a supplement, and which types of supplements) and continuous variables (for example, fertiliser and irrigation levels, the length of the mating period, planting date, and culling proportions). Whether the discrete and continuous variables are coded to a fine-grained binary representation, or whether these are all represented by continuous variables (where the discrete options are based on a 'closest' value decision rule) is largely up to the user. As evident in Table 2 of Chapter 5, historical applications have favoured binary applications, but these may well have been influenced by the wider availability (particularly in the English language) of genetic algorithm publications in that era. Interestingly, Campbell et al. (1998) started out with a binary genetic algorithm, and then progressed to one with real-value representation. Other examples of real-value coding are also listed in this table. Our series of studies (Mayer et al. 1997, 1999a, 2001) have used both binary and real-value representation, on the same problem (a large beef breeding model), both with success.

In practice, there appears little difference in the performance of these two basic methods of coding (Mühlenbein and Schlierkamp-Voosen 1994), and by extension, any other representation should also work well. It is the effect of other key operational parameters on the chosen representation that really

drives the evolutionary algorithm. Provided robust values of these are used, the exact choice of coding appears less critical.

POPULATION SIZE

The number of members in the population is one of the most important choices that designers of evolutionary algorithms have to make (Michalewicz and Fogel 2000). Whilst adaptive sizes have been suggested, a constant population size is generally used, probably more because of convenience (Spears 2000). In determining this number, a balance must be obtained – a population which is too small will not contain the necessary genetic diversity required for the proper functioning of the evolutionary algorithm, but excessively large populations will slow the optimisation due to inefficiencies. As Saloman (1996) also notes, larger populations may be needed when the problem has interactions between the input options (epistasis).

A number of theoretical studies have investigated the effect of population size (Goldberg 1989, Goldberg et al. 1992, Peck and Dhawan 1995, Smith 1997, Pelikan et al. 2000, Pelikan and Goldberg 2001), and developed equations for estimating the optimal population size. However, these studies mostly investigated simpler problems with known properties, and their extension to larger, unknown models is speculative (Michalewicz and Fogel 2000). With real-world problems, few users will have any idea of the necessary values for the variables in these formulae, such as the degree of difference between the average fitness values of different schemata, or the variance of the building blocks. One common result which can be used, however, is that the necessary population size is directly related to the problem size.

An early rule of thumb from the binary genetic algorithms field is to take the population size as equal to, or slightly larger than, the number of bits contained in the representation of each member. This appears to work in practice – on the crop and irrigation scheduling model with 49 bits, Kuo et al. (2000) found a population size of 50 to be superior to both 30 and 100. On a 16-dimensional sheep breeding model with an unspecified number of bits (which, given the complexity of the coded management options, is likely to be towards 100), Horton (1996) showed a population of 50 members to be insufficient – these optimisations tended to converge to local optima. A population of 100 members was required here. Similarly, Hart et al. (1998) outlined a 15-dimensional dairy farm model (again, with an unspecified number of bits) upon which a population size of 10 proved inferior, with 50

required. On their simpler version of the model (6 dimensions), a population of 20 members proved adequate.

Our series of modelling investigations using binary genetic algorithms have tended to be less conclusive – on the dairy farm model with 77 bits, approximately equal performance was obtained from populations of 40 and 80 members (Mayer et al. 1996a). Similarly, population sizes of 30 and 50 gave similar results and rates of convergence on the 20-dimensional beef model with around 80 bits (Mayer et al. 1999b). On the more difficult version of this beef model (40 dimensions and around 120 bits), for a given total number of model runs, approximately equal performance was obtained from populations of 30, 50, 100 and 150 members. Hence, the presence of more generations appears capable of making up any deficiencies of small population sizes. One counter-example is given in Parmar et al. (1996) – for a fixed number of runs (400) on a model with 6 dimensions and 12 bits, a population of 50 proved superior to 25 and 40. However, this result was obtained under 50% truncation selection, so maybe under this more extreme selection pressure higher population numbers were required to maintain genetic diversity.

On agricultural problems with real-value coding, quite different relative population sizes have been used with success, indicating that perhaps this parameter is not so critical. High values were used in Meszaros et al. (1999) (populations of 300 to 1000 for a 9-dimensional problem), Cho and Lee (2000) (500 for an 8-dimensional model), and Davies et al. (2000) (5 subpopulations each having 50 members, for an 8-dimensional problem). On a 51-dimensional greenhouse model, Polheim and Heißner (1997) reported success of both a genetic algorithm with 4 subpopulations of 50 members, and an evolution strategy with 3 subpopulations of 2 members. Mardle and Pascoe (2000) found the optimum of a very large problem (876 dimensions) using only a relatively small population size of 70. Some comparisons of population sizes come from our studies of the beef property model – with 40 dimensions (Mayer et al. 1999a), populations of 100 and 500 members appear better than 50, as exemplified in Figure 1 (from Mayer 2000), although these results are somewhat confounded by changes in the other key operational parameters. On the 70-dimension version of this model (Mayer et al. 2001), populations of 200 members were consistently superior (being more optimal at all stages of the searches) than optimisations using a population size of 500. Hence, overall it appears that for real-value coding a 'good' population size will be at least greater than the number of parameters being optimised, but also that excessively large sizes do penalise progress.

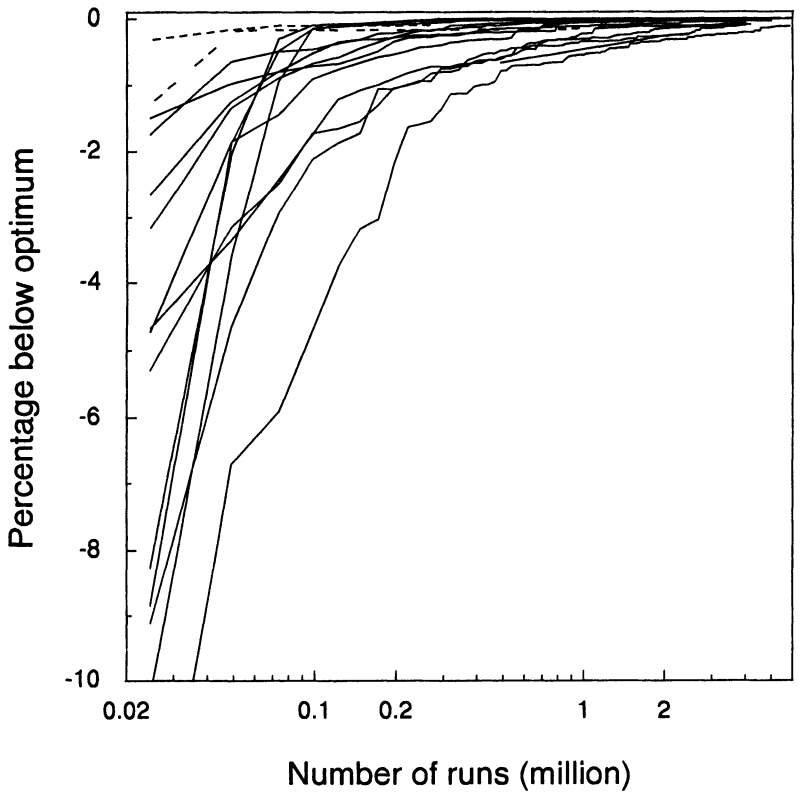

Figure 1. Rates of convergence for (real-value) genetic algorithm optimisations of the beef property model with 40 dimensions. Solid lines are for populations of 50 members, dashed for 100, and dashed with gaps for 500.

As well as choice of the overall population size, practitioners also need to consider the use of subpopulation methods to prevent premature convergence to a local optimum – these methods have also been termed sharing, deming, niching, speciation, and islanding methods (Goldberg 1987, Goldberg 1989, Davis 1991b, Spears 2000). Here, extra detailed coding is required, covering the definition and number of subpopulations, restrictions on mating, competition between subpopulations, similarity metrics, sharing, and migration. These methods have been shown to be effective in test cases (mostly test functions), which are obviously of 'known' form and shape. Here, the number and size of the subpopulations can be aligned with the expected number of optima, thus preventing under- or over-crowding (Michalewicz and Fogel 2000). Considering agricultural models, Polheim and Heißner (1997) and Davies et al. (2000) used subpopulation methods, but without any 'whole-population' optimisation for comparison, the success

of this method cannot be judged. Determining a general subpopulations structure with the expectation of robustness across different and unknown problem types would appear daunting, and very difficult to demonstrate.

One further note is that, in keeping with a number of operational parameters for evolutionary algorithms, population size cannot be considered in isolation (Smith 1997). Its effect on the progress of each optimisation interacts with the other key parameters - Schaffer et al. (1989) found a significant three-way interaction between population size, mutation rate, and recombination rate. A 'seemingly poor' population size (either too large or too small) can apparently be compensated for by appropriate choices of the other parameters. Overall, following the basic rules of thumb seems prudent – take a population size of either slightly larger than the number of binary bits, or somewhat larger than the number of real-value genes. As there are distinct disadvantages of having too small a population, but not much disadvantage to having a few too many population members, a safe choice would be to adjust the number upwards in doubtful situations.

SELECTION OF PARENTS

As evidenced by its key role in evolutionary programming and its potential for controlling the degree of exploration versus exploitation, selection can be used as one of the main driving mechanisms of evolutionary algorithms. The range of possible operators are well described in texts (Davis 1991b, Michalewicz 1996, Bäck et al. 1997a), and include fitness proportional (also called Roulette-wheel, where each population member is allocated a selection probability based on its fitness value), scaled fitness proportional (where the fitness values are linearly or nonlinearly scaled prior to the allocation of selection probabilities – this also solves the problem of any negative fitness values), ranked proportional (similar, except the probabilities are based on the rankings of fitness), truncation (where only a chosen percentage of the population is used), Queen bee (the elite population member is always used as the first parent in each mating), and tournament (two or more members are taken at random, and only the best of these used as a parent).

Each of the selection techniques can be used to exert more or less selection pressure, as required by the user. Obviously, a smaller percentage in truncation selection will focus the search around the elite individuals. The allocated probabilities under any of the proportional selection schemes can be used to exert greater or lesser selection pressure. Under tournament selection, larger tournament sizes result in more pressure, as it is more likely that the top value each time will be from near the top of the list – for

example, with n individuals being chosen for each tournament, the poorest (n-1) members can never be chosen as a parent. Blickle (1997) demonstrated a relatively high loss of diversity (and hence high selection pressure) when using $n = 5$.

As Hancock (1997) notes, the concept of selection can be used to somewhat bridge the apparent gap between evolution strategies and genetic algorithms. Each generation, an evolution strategy generates 'many' offspring, with only the best of these then taken forward to the next generation, and used as parents. Genetic algorithms traditionally generate fewer offspring, and (proportionally) take more forward – but if truncation or a high-pressure selection method is used, the next parents will generally be from the 'top portion' of the members, which roughly equates to the evolution strategy method.

A number of theoretical studies have demonstrated the approximate equivalence of some of the selection techniques – different implementations of tournament, truncation, and ranked proportional selection can give the same expected number of offspring per parent (De Jong and Sarma 1995, Blickle and Thiele 1997). From a practical viewpoint, it has long been acknowledged (Goldberg 1989) that fitness proportional selection can have problems with a 'bad' distribution of fitness values (especially early in the search, when one or more 'good' population members stand out). Appropriate scaling, or ranking, of the fitness values can solve this. Under any of these Roulette-wheel selection schemes, the fitness values have to be summed, and the selection probabilities reallocated to individuals, each time the population membership is changed. This is no problem if generational replacement is used, but under steady-state replacement (which is now usually recommended), this reallocation is required with the addition of each new population member. From a computational viewpoint, this is clearly inefficient. Conversely, tournament selection does not have this overhead, and can easily be implemented with steady-state replacement.

Applications to agricultural systems models illustrate the wide range of possible selection methods (Appendix 1). Here, the proportional selection methods have tended to dominate historically, despite the recent trend towards these becoming generally less common in the wider evolutionary algorithm field (Michalewicz and Fogel 2000). With binary genetic algorithms, Parsons (1998) found the performance of scaled fitness proportional and ranked proportional selection to be approximately equal, whereas in Mayer et al. (1999b) the ranked version was consistently inferior. Tournament selection, usually with low tournament sizes (two or three), has also been used successfully (Appendix 1).

Overall, results suggest that choice of selection method is not too critical, as all of the available options appear to work quite well in practice.

Considering computational efficiency and results from the wider evolutionary algorithms literature, tournament selection (with a tournament size of two) is recommended. If more selection pressure is desired, then truncation selection or a larger tournament size could be used.

REPLACEMENT STRATEGY

Historically, generational replacement has been widely used. Under the evolution strategy approach, a population with μ members generates λ offspring (where $\lambda > \mu$). These offspring are then available for the next generation – either to totally replace the original parents under the (μ,λ) scheme, or to compete with the parents for placement in the next generation $(\mu+\lambda)$. The original form of genetic algorithms generates as many offspring as there are members in the population, with these then replacing the parents to form a new generation (this is termed a generation gap of one) – which thus equates to a (μ,μ) scheme. Here, the use of elitism (where, at least, the best parent is automatically retained) was strongly recommended (Jones 1995, Michalewicz 1996).

More recently, steady-state replacement of population members has become more the norm, with only the poorest individuals (or single individual) of the population being replaced at each step. This ensures the retention of the best parents, and also makes the new offspring immediately available to the optimisation (which is advantageous, as they should be amongst the 'best' individuals at any time). The question of whether to exclude duplicates also needs to be considered – obviously, a computational overhead is required to test if offspring are direct copies of any parent, with this test having possible resolution difficulties with real-value codings (i.e., how close do the values have to be to be considered equal?). Also, this may well interfere near the end of the optimisation, where most population members are expected to converge to the globally optimal solution. As the inclusion of duplicates does not appear to penalise performance, most evolutionary algorithms merely accept them.

When adding the new individuals to the population, the replacement of the current members can be deterministic (where the worst, or lowest-fitness, individuals are automatically deleted), or stochastic (where some form of random rules are used to determine which population members are replaced). As the latter method can result in the retention of some of the poorest individuals, at the possible expense of the best or the 'elite' members of the population, it appears a less optimal method (unless the user has a specific reason for adopting this). Hence, steady-state (continuous) and deterministic

replacement of the worst population members by the best generated offspring is recommended.

MUTATION

Mutation has a number of different roles in the progress of an optimisation. Early on, it allows a degree of exploration as some of the model's options are changed to different values. In the mid stages, it provides an escape mechanism, for optimisations which may have converged (largely through recombination) to a local optimum. In the later stages, mutation helps the convergence of each option onto the global optimum.

This operator was always used as the key driver of the evolution strategies approach, whereas under the traditional genetic algorithms it was perceived as only a background operator. In the latter, only small mutation rates (around 0.001 to 0.01) were used, although under binary representations the rate of mutation of any single option would be higher, as each of these options generally maps onto a number of bits. More recently, under the more generic evolutionary algorithms approach, much higher rates have been used (Hinterding et al. 1995, Michalewicz 1996, Bäck 1997). Morimoto and Hashimoto (1996) point to recent studies in molecular biology to justify their rather extreme mutation rate of 0.8. Higher values obviously tend to be quite disruptive of existing schemata (Spears 2000), and can interfere with convergence. Variable mutation rates have been proposed, but these have tended to give only marginal improvements (Mühlenbein and Schlierkamp-Voosen 1994). Of course, the evolution strategies method carries self-adapting standard deviations which control the degree of mutation as the optimisation progresses, but this is at the expense of the doubling of the effective parameter space.

Some studies have been conducted on mutation rates, showing the optimal rate to be largely problem-dependent (Hinterding et al. 1995, Michalewicz 1996, Bäck et al. 1997b). For real-value representation, a lower bound is $1/n$ (Bäck et al. 1997b), where n is the number of genes or coded model options. At this value, the expectation is for one option to be mutated per offspring. This has been recommended as a reasonable heuristic for the range of unknown problems (thus including systems models), for which it is impossible to derive analytical results (Bäck 1997). Alternately, Saloman (1996) contends that more than one gene needs to be mutated per offspring, for effective progress in real-world cases. On test functions, Hinterding et al. (1995) found $5/n$ to be optimal.

Regardless of the chosen mutation rate, the available methods of implementing mutation also vary somewhat. For binary representation, the

mutation of each bit is a straight-forward bitflip – changing the value to its complement. For higher-level representations, a range of methods exists. Boundary mutation shifts the mutated option to either its upper of lower boundary (randomly determined), which can be useful for models where this may be expected for the optimal solution. Uniform random mutation changes the value to a point chosen randomly (within the specified bounds). More common, however, are the 'centre-heavy' methods, where small shifts in the value have higher probability, although larger shifts are possible. These include the delta (triangular) method, and from the statistical field use of the Normal or Gaussian distribution, along with the Cauchy (which is similar, but has comparatively fatter tails). The width of these distributions (as usually specified by their standard deviation or variance) can be fixed, or be included as self-adapting parameters in the optimisation (under an evolution strategy approach).

In practice, widely differing types and rates of mutation all tend to work well with agricultural models, as exemplified by the ranges used in Appendix 1. Direct comparisons here are few, and give few insights. In a binary genetic algorithm, low-level 'background' mutation rates of 0.001, 0.005 and 0.01 all performed similarly (Mayer et al. 1999b). On the same model with real-value representation, a variety of mutation types was trialed, with rates of up to 0.25, all with success. Using an evolution strategies approach, the only real failure noted (being 1.5% short of the global optimum) was with nil mutation (using recombination only). Any mutation (rates here varied between 0.001 and 0.6) generally worked (Mayer et al. 1999b). One interesting result here was that 'double mutation' performed best. This approach used lower-level Gaussian mutation (at a probability of 0.05) combined with higher-level boundary mutation (probability of 0.20). However, extending this multiple mutation to 12 different combinations of both types (Gaussian, boundary and uniform random) and rates (ranging from 0.02 to 0.50) gave poorer performance than the basic double-mutation combination (Mayer et al. 2001). These results, however, may be particular to the model studied, which had the optimal solution with a number of its management options on their respective boundaries.

Overall, a wide range of mutation types and rates have been shown to work well. Again, this parameter cannot be considered in isolation from the other operational parameters of the evolutionary algorithm. It can be used as the primary driver of exploration and fine-tuning, or merely as a background insurance-type operator. The use of double or multiple mutation operators appears to offer some promise.

RECOMBINATION

In the original binary genetic algorithms, recombination (alternately termed crossover) was considered the main driving force. Here, high recombination rates (being the probability that recombination is applied) were traditionally used (Goldberg 1989). Recombination has also proven to be one of the most important with the more recent generic evolutionary algorithms. Michalewicz and Fogel (2000) question whether the high rates used for binary representations necessarily flow across to alternate codings. However, good results have been obtained across virtually the whole range of rates (0 to 1) (Park 1995), so this value of the recombination rate appears less critical.

Recombination can take a number of very different forms, all of which provide a mixing of the genetic material of the selected parents. The simplest (and original) was a one-point crossover, where the parents' genes were crossed at one randomly-chosen point. This concept logically extends to two-point, or three, or up to n-point crossover, where n is the number of coded genes. Here, a random choice is made between the parents for each gene, and this method is also termed uniform crossover (Saloman 1996). Also available is majority-logic recombination (Michalewicz and Fogel 2000), where the dominant gene from more than two parents is chosen. All these forms of recombination can be applied with either binary or alternate representations, as for each gene they effectively take the value of one of the parents.

Under real-value codings, other recombination operators are also available. Intermediate arithmetical recombination results in a randomly-chosen value which lies between the values of the two parents. This tends to be a contractive operator (subsequent generations will tend towards the mean), and if used needs to be balanced by an expansive mutation operator. Alternately, extended arithmetical crossover allows values outside the range of the parents (up to the specified bounds), so is an expansive version of this operator. Also available (but little used) is simplex recombination, where the simplex algorithm (Nelder and Mead 1965; see Chapter 4) is applied with more than two parents, to determine the new value of each gene.

Again, the form of the chosen recombination operator can be used to balance exploration versus exploitation. Those which cause minimal disruption of the parents' values (for example, one-point crossover) tend towards exploitation, whereas the more disruptive forms such as extended arithmetical recombination lean heavily towards exploration. In theoretical analyses, Spears (2000) has proven what is logically apparent – the more disruptive forms of recombination increase the rate of destruction of existing 'good' schemata, but balance this by increasing the rate of construction of

new schemata. Hence, the optimal type and rate of recombination will be problem-dependent.

As with many evolutionary algorithm operators, comparisons between the different forms have met with somewhat mixed results. On a suite of test functions, Schaffer et al. (1989) demonstrated two-point crossover to be marginally better than one-point. With network designing, Rothlauf et al. (2000) found uniform and one-point crossover to perform equally well, within both genetic algorithm and evolution strategy frameworks. On a silviculture harvesting problem, Lu and Eriksson (2000) suggested two-point crossover at a rate of 0.9 during the search (exploration) phase, and one-point at 0.5 for fine-tuning. Our series of optimisations of a real-value coded beef model (Mayer et al. 1999b), also across genetic algorithm and evolution strategy frameworks, pointed to a slight advantage of uniform and one-point crossover over extended arithmetical recombination.

Whilst Jones (1995) contends that recombination is not advantageous on all problems, Fogel (1995b) and others maintain that recombination and mutation (in particular) have a synergistic effect, and always should be used in combination. Spears (1997) also shows that the effect of recombination interacts with population size – high recombination rates are required for small populations (and vise-versa), whilst the whole range of rates works well with medium population sizes. In an early recommendation which still appears appropriate, South et al. (1993) suggest a rate of 0.95 with a smaller population for fast optimisation, and 0.45 with a large population for a slower and more thorough operation. Overall, a robust choice would appear to be a moderate to high rate (around 0.80) with a 'medium-disruptive' type such as uniform.

OVERALL

As has repeatedly been demonstrated, and in line with the theorem of Wolpert and Macready (1997), the range of operational parameters will have varying effects, dependent mainly on the problem type (Goldberg 1989, Davis 1991b, Fogel 1995a, Horton 1996, Michalewicz 1996). Drawing all these parameters together into a robust 'best-bet' set at first appears difficult (Michalewicz and Fogel 2000), and yet there are some areas of good agreement.

Firstly, the historical 'great debate' of which form of evolutionary algorithm is best is no longer an issue. Whilst the genetic algorithms style tends to be better at global searches, and evolution strategies superior for local (Polheim and Heißner 1997), any form which uses selection, mutation and recombination conforms to the basic evolutionary doctrine, and should

be effective (Mühlenbein and Schlierkamp-Voosen 1993, Beyer 1997). These key operators have a synergistic effect, and algorithms which do not include all three have proved inferior (Bäck and Schwefel 1993, Michalewicz 1996). They also have differing degrees of effect as the optimisation progresses – through heritability studies, Mühlenbein and Schlierkamp-Voosen (1994) demonstrate how evolutionary algorithms are initially driven by recombination, and then by mutation near the end.

Another decision to be considered is whether to use the key operators of mutation and recombination together in the generation of each offspring, or whether to use these in parallel (where each new member is generated either by mutation, under an evolution-strategies type method, or by recombination, as per genetic algorithms). The former is the traditional approach, with the latter as suggested by Davis (1991b) and used in some implementations - for example, Genial (Widell 1997). The separate application of these operators has the advantage of allowing them to have different levels of influence as the optimisation proceeds. Of course, when a member produced here by recombination subsequently undergoes mutation (say, in the next generation), this equates to the simultaneous application of both operators. Hence, this choice of approach is perhaps not so critical.

For the effective solution of future unknown problems, a consolidation of all results leads us to recommend the following as a likely robust set of operational parameters for evolutionary algorithms. Of course, different users can and will experiment with alternate values and options, to find those most suited to their problem.

- Coding of the modelled options to genetic representation. Either binary or real-value coding can be effective in practice. The choice largely depends on which is more aligned with the problem on hand, as well as the capabilities of the chosen evolutionary algorithm.
- Population size. Populations which are too small tend to be suboptimal, whereas an overly-large population does not appear to penalise the optimisation much. Hence, the 'safe' options of a population size greater than the population members' number of bits (for binary representation), or around twice the dimensionality of the problem (for real-value representation), should be used.
- Selection of parents. Tournament selection is recommended, as it is equivalent to most of the available methods, but computationally more efficient. A tournament size of two appears a good default, but higher values can be used if more selection pressure is desired.
- Replacement strategy. The deterministic replacement of the worst population members by the best generated offspring, on a steady state (continuous) basis, appears best.

- Mutation. Good performances from a wide range of mutation types and rates have been demonstrated. Robust options are lower-level rates (around 0.01 to 0.05) of bitflip for binary genetic algorithms, and Gaussian mutation for real-value coding, at a rate of at least the inverse of the number of genes. The simultaneous use of two different mutation operators shows promise.
- Recombination. Studies suggest that a moderate to high rate (around 0.8), using a 'medium-disruptive' type such as uniform, should work well across a range of problem types.

Chapter 7

FUTURE DEVELOPMENTS

Evolutionary algorithms are continually being adapted and enhanced, in a wide range of disciplines. Both theoretical and applied publications appear regularly, and conferences in a number of evolutionary themes continue. Most obviously, theoretical studies are always useful, as existing theories on just how evolutionary algorithms work (particularly with regard to the interactions between operators) are currently unsatisfactory (Beyer 1997). These theoretical developments need to both expand the formal mathematical basis for the estimation of optimal parameters, and to justify heuristic results (Michalewicz and Fogel 2000).

From a practical point of view, a number of future analyses and approaches appear to have merit. Spears (2000) suggests that the range of different problems be categorised according to generic type, and that optimal parameters for evolutionary algorithms be found for each of these types. This task would be lengthy and difficult, but if achieved would give users a suite of well-tuned algorithms for use on their particular problem.

Schwefel (1997) contends that further progress in evolutionary algorithms will be made by reverting back closer to nature. In particular, multiple selection criteria (to mimic multiple predators or dangers), diploid representation with dominance and recessiveness rules, and simulating sexes and mating behaviour are all suggested here.

Researchers are continually finding improvements for evolutionary algorithms, ranging from minor fine-tunings to breakthrough methods. Some of these are particular to the systems or problems being studied, and some will have wider application. The list of possible developments currently under investigation includes alternate and improved operators and codings, self-adapting mechanisms, subpopulation methods, the

incorporation of hybrids and heuristics, and Bayesian operators (Pelikan et al. 2000).

Historically, systems researchers in the agricultural field have tended to be 'late adopters' of evolutionary algorithm developments. Under this conservative approach, the new techniques (here, genetic and evolutionary algorithms) need to be well proven in other fields before being widely adopted. This 'wait and see' approach avoids the possible errors and pitfalls of unproven developing technology, but also imposes a penalty of not using the best methods as they become available. It is hard to see this changing in the future, although the challenge of optimising larger and more difficult systems models may well force agricultural researchers to use the most up-to-date evolutionary methods.

APPENDIX 1

Applications of evolutionary algorithms to agricultural systems.

Reference	Problem type	Type of EA[#]	Objective function	Search space	No. of params.
Annevelink (1992)	Spatial allocation in greenhouse production	Binary GA	Gross margin of operation		
Arias et al. (1998)	Climate control for greenhouse lettuces	Real–value GA	Total heating costs		
Barioni et al. (1999a)	Grazing and fertilizer management of NZ sheep farms	GA	Farm gross margin		22
Cacho and Simmons (1999)	Multi-year farm investment strategies	Binary GA	Farm gross margin	10^6	2
Campbell et al. (1998)	Fitting model parameters to data for pesticides in wool	Real–value, binary GAs	ln(Residual sum of squares)		42
Cho and Lee (2000)	Route optimisation for autonomous orchard sprayer	Integer GA	Residual mean square		8
Davies et al. (2000)	Experimental design for silage improvement trials	Integer GA	Nonlinear function of costs and biochem. parameters	1.7×10^6	8
Franchini (1996, 1998)	Calibration of model parameters	Real–value GA	Residual sum of squares	10^{23} – 10^{33}	11 – 13
Goggos and King (2000)	Dynamic temperature control for greenhouse production	Binary GA	Absolute error (integrated over time)		
Hart et al. (1998)	Herd and paddock management of a dairy farm	Binary GA	Farm's milkfat production	10^{33}	6, 15
Hayes et al. (1997)	Mate selection in controlled breeding program	Binary GA	Estimated breeding values		4 – 20

[#]EA: Evolutionary Algorithm; GA: Genetic Algorithm; ES: Evolution Strategy

Popn. size	Gener-ations	Repli-cates	Parent selection	Replac. strategy	Recombination Type	Recombination Probability	Mutation Type	Mutation Probability
			Roulette-wheel	Elitism	1-point		Bitflip	
10 – 20	150		Roulette-wheel	Elitism	1-point	0.90	Gaussian	0.15
40	25					0.60		0.01
100	200	100	Roulette-wheel (scaled scores)	Generat-ional replace-ment	1-point	0.60	Bitflip	0.01
80 – 150	80 – 100							
500			Tourn-ament			0.70		0.05
50	5		Roulette-wheel (scores)	$(\mu+\lambda)$	$\geq$ 1-point	0.60		
125 – 500	40	10	Roulette-wheel (ranks)	Elitism	2-point		Random	0.01
			Roulette-wheel (scores)	Elitism				
10,20, 50	1000	10	Tourn-ament (2 or 3)		2-point, uniform		Bitflip	
5 – 30	8 – 10^5						Bitflip	$\leq$ 0.40

Reference	Problem type	Type of EA[#]	Objective function	Search space	No. of params.
Horton (1996)	Genetic improvement in Australian sheep-breeding industry	Binary GA	Expected economic gain	10^{16}	16
Husmann and Tantau (2001)	Energy supply for floriculture greenhouses	Binary GA	Total annual heating cost		≥ 6
Kozan (1999)	Dispatch processes in pot-plant nursery	Discrete-value GA	Total distance covered	10^{19}	21
Kuo et al. (2000)	Irrigation, area and crop scheduling	Binary GA	Region's gross margin	10^{15}	14
Lu and Eriksson (2000)	Harvest scheduling of forestry stands	Binary GA	Net present value		650
Mardle and Pascoe (2000)	Management of English Channel fisheries	Real and integer GA	Fisheries profit		876
Mariano (1998)	Water distribution networks for irrigation	Binary GA	Network cost	10^{9}	8
Mayer et al. (1995, 1996a)	Management of a dairy farm	Binary GA	Farm gross margin	10^{23}	16
Mayer et al. (1999a)	Trading (buying and selling) strategies for beef breeding property	Binary GA	10-year net present value	10^{50}, 10^{100}	20, 40
Mayer et al. (1999b)	Trading (buying and selling) strategies for beef breeding property	Binary GA	10-year net present value	10^{100}	40

[#]EA: Evolutionary Algorithm; GA: Genetic Algorithm; ES: Evolution Strategy

Popn. size	Gener-ations	Repli-cates	Parent selection	Replac. strategy	Recombination Type	Recombination Probability	Mutation Type	Mutation Probability
50 – 100	120	10	Roulette -wheel (scaled scores)	Elitism – best 4 retained	10-point	1.00	Bitflip	0.20
30	60						Bitflip	
			Roulette-wheel (scores)		2-point	1.00	Node-swap	
30 – 100	100 – 800		Roulette-wheel (scores)	Generat -ional replace -ment	1-point	0.60	Bitflip	0.02
200	1000 – 2000		Roulette-wheel (ranks)		1-point, 2-point	0.50 – 0.90	Bitflip	0.001 – 0.02, dynamic
70	10^5		Roulette-wheel	Elitism	n-point			
			Roulette-wheel (ranks)		Uniform	0.60 – 1.00	Bitflip	0.01 – 0.10
40, 80	125, 250	48	Roulette-wheel (scores, ranks)	Elitism	1-point	0.45 – 0.95	Bitflip	0.001, 0.01
30, 50	4×10^4 – 10^6	9	Roulette-wheel (scores, ranks)	Elitism	1-point	0.60	Bitflip	0.001, 0.01
30 – 150	3×10^4 – 2×10^5	8	Roulette-wheel (scores, ranks)	Elitism	1-point	0.40 – 0.80	Bitflip	0.001 – 0.01

Reference	Problem type	Type of EA#	Objective function	Search space	No. of params.
Mayer et al. (1999b)	Trading (buying and selling) strategies for beef breeding property	Real-value GA ES	10-year net present value	10^{100}	40
Mayer et al. (2001)	Whole-property management for beef breeding property	Real-value GA	10-year net present value	10^{120}	70
Meszaros et al. (1999)	Genetic improvement of sheep flocks	Real-value GA	Cumulative genetic gain over 20 years	10^{15}	9
(Moore et al. (2000)	Silviculture harvest scheduling with preservation of wildlife	Trinary GA	Probability of extinction	10^{13}	40
Morimoto and Hashimoto (1996)	Growth control in hydroponic tomatoes	Binary GA	Leaf length ÷ stem diameter (indicator of yield)		4
Pabico et al. (1999)	Determining cultivar coefficients of crop model	Binary GA	Scaled L_1–metric distance	10^{32}	13
Parmar et al. (1996)	Machinery selection for peanut farming	Binary GA	Net returns above costs	4100	6
Parsons (1998)	Silage harvesting scheduling	Binary GA	Total feeding costs	10^{9}	21

#EA: Evolutionary Algorithm; GA: Genetic Algorithm; ES: Evolution Strategy

Popn. size	Gener-ations	Repli-cates	Parent selection	Replac. strategy	Recombination Type	Recombination Probability	Mutation Type	Mutation Probability
50 – 500	10^4 – 10^5	15	Roulette-wheel, Tourn.(2)	Elitism	Various	0.25 – 0.50	Various	0.001 – 0.25
20 – 50	2×10^5 – 5×10^5	13	Strictly random	(μ+λ)	Various	0 – 0.33	Various	0 – 0.60
200, 500	127 – 317	4	Tourn-ament (2)	Sub-generat-ional - 80% replaced	Various	0.50 – 1.00	Various	0.02 – 0.50
300 – 1000	200			Sub-generat-ional - 50% replaced	1-point		Dynamic	≈ 0.20
200	200	20	Roulette-wheel (ranks)	Elitism	2-point	0.80	Random choice (of 3)	0.10
6	30		Strictly random	Elitism	1-point	0.80	Single bit flipped	0.80
30	2000	5	Roulette-wheel		1-point	0.60	Bitflip	0.033
15 – 50	4 – 14	10	Roulette -wheel (scores)	5% to 100% replaced	1-point	0.60	Bitflip	0.10
10, 20	40	10	Roulette-wheel (ranks)	Elitism	1-point		Bitflip	0.10

Reference	Problem type	Type of EA[#]	Objective function	Search space	No. of params.
Polheim and Heißner (1997)	Climate control for horticultural greenhouses	Real-value GA and ES	Profit (value of produce, less costs)		51
Verryn and Roux (1998)	Selection in tree breeding scheme	Binary GA	Realised genetic gain		
Wang (1991)	Calibration of model parameters	Binary GA	Residual variance	10^{21}	7
Woodward, S.J.R. (pers. comm., 2000)	Grazing management for a New Zealand sheep farm	Binary GA	Farm gross margin		8

[#]EA: Evolutionary Algorithm; GA: Genetic Algorithm; ES: Evolution Strategy

Popn. size	Gener-ations	Repli-cates	Parent selection	Replac. strategy	Recombination Type	Recombination Probability	Mutation Type	Mutation Probability
200			Roulette -wheel (ranks)	Sub-generat-ional - 90% replaced	discrete and line		ES-style	0.02
30 – 100			Roulette-wheel (scaled scores)				Bitflip	
100	50	10	Roulette-wheel (ranks)	Generat -ional replace -ment	2-point	1.00	Bitflip	0.01
50	60					0.60	Bitflip	0.10

REFERENCES

Ackley, D.H. (1987). An empirical study of bit vector function optimization. In *Genetic algorithms and simulated annealing* (Ed, L. Davis), Pitman, London, 170-204.

Allen, P.M. and McGlade, J.M. (1986). Dynamics of discovery and exploitation: The case of the Scotian Shelf groundfish fisheries. *Canadian Journal of Fisheries and Aquatic Science* **43,** 1187-1200.

Allen, P.M. and McGlade, J.M. (1987). Modelling complex human systems: A fisheries example. *European Journal of Operational Research* **30,** 147-167.

Anderson, J.R., Dillon, J.L. and Hardaker, J.B. (1977). *Agricultural decision analysis.* Iowa State University Press, Ames.

Angeline, P.J. and Fogel, D.B. (1997). Other representations. In *Handbook of evolutionary computation* (Eds, T. Bäck, D.B. Fogel and Z. Michalewicz), Institute of Physics Publishing, Bristol, C1.8:1-2.

Annevelink, E. (1992). Operational planning in horticulture: Optimal space allocation in pot-plant nurseries using heuristic techniques. *Journal of Agricultural Engineering Research* **51,** 167-177.

Anonymous (1972). A new algorithm for optimization. *Mathematical Programming* **3,** 124-128.

Arias, A.R., Goddard, J. and Cruz, I.L. (1998). A hybrid system for optimal greenhouse climate control using artificial intelligence techniques. *Proceedings Seventh International Conference on Computers in Agriculture,* 26-30 October 1998, Hyatt Orlando, 780-786.

Bäck, T. (1997). Mutation parameters. In *Handbook of evolutionary computation* (Eds, T. Bäck, D.B. Fogel and Z. Michalewicz), Institute of Physics Publishing, Bristol, E1.2:1-7.

Bäck, T., Fogel, D.B. and Michalewicz, Z. (Eds.) (1997a). *Handbook of evolutionary computation.* Institute of Physics Publishing, Bristol.

Bäck, T., Fogel, D.B., Whitley, D. and Angeline, P.J. (1997b). Mutation. In *Handbook of evolutionary computation* (Eds, T. Bäck, D.B. Fogel and Z. Michalewicz), Institute of Physics Publishing, Bristol, C3.2:1-8.

Bäck, T., Hammel, U. and Schwefel, H.-P. (1997c). Evolutionary computation: Comments on the history and current state. *IEEE Transactions on Evolutionary Computation* **1,** 3-17.

Bäck, T. and Schwefel, H.-P. (1993). An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation* **1,** 1-23.

Balinski, M.L. and Wolfe, P. (1975). *Nondifferentiable optimization.* North-Holland, Amsterdam.

Barabino, G.P., Barabino, G.S., Bianco, B. and Marchesi, M. (1980). A study on the performances of simplex methods for function minimization. *Proceedings IEEE International Conference on Circuits and Computers,* New York, 1150 - 1153.

Bard, Y. (1974). *Nonlinear parameter estimation.* Academic, New York.

Barioni, L.G., Dake, C.K.G. and Parker, W.J. (1999a). Optimizing rotational grazing in sheep management systems. *Environment International* **25,** 819-825.

Barioni, L.G., Tedeschi, L.O. and Lanna, D.P.D. (1999b). A combination of linear and nonlinear optimisation algorithms to maximise net return in feedlots. *Proceedings International Congress on Modelling and Simulation,* December 1999, The University of Waikato, Hamilton, 807-812.

Battiti, R. and Tecchiolli, G. (1994a). The reactive tabu search. *ORSA Journal on Computing* **6,** 126-140.

Battiti, R. and Tecchiolli, G. (1994b). Simulated annealing and tabu search in the long run: A comparison on QAP tasks. *Computers and Mathematics with Applications* **28,** 1-8.

Bennett, D. and MacPherson, D.K. (1985). Structuring a successful modelling activity. *ACIAR Proceedings* **11,** 70-76.

Bertsekas, D.P. (1975). Nondifferentiable optimization via approximation. In *Nondifferentiable optimization* (Eds, M.L. Balinski and P. Wolfe), North-Holland, Amsterdam.

Bertsekas, D.P. and Tsitsiklis, J.N. (1989). *Parallel and distributed computation.* Prentice Hall, New Jersey.

Beyer, H.-G. (1997). An alternative explanation for the manner in which genetic algorithms operate. *BioSystems* **41,** 1-15.

Blickle, T. (1997). Tournament selection. In *Handbook of evolutionary computation* (Eds, T. Bäck, D.B. Fogel and Z. Michalewicz), Institute of Physics Publishing, Bristol, C2.3:1-4.

Blickle, T. and Thiele, L. (1997). A comparison of selection schemes used in evolutionary algorithms. *Evolutionary Computation* **4,** 361-394.

Bohachevsky, I.O., Johnson, M.E. and Stein, M.L. (1986). Generalized simulated annealing for function optimization. *Technometrics* **28,** 209-217.

Bos, J. (1993). Zoning in forest management: A quadratic assignment problem solved by simulated annealing. *Journal of Environmental Management* **37,** 127-145.

Botes, J.H.F., Bosch, D.J. and Oosthuizen, L.K. (1996). A simulation and optimization approach for evaluating irrigation information. *Agricultural Systems* **51,** 165-183.

Box, M.J. (1965). A new method of constrained optimization and a comparison with other methods. *The Computer Journal* **8,** 42-52.

Bramlette, M.F. and Bouchard, E.E. (1991). Genetic algorithms in parametric design of aircraft. In *Handbook of genetic algorithms* (Ed, L. Davis), Van Nostrand Reinhold, New York, 109-123.

Bramlette, M.F. and Cusic, R. (1989). A comparative evaluation of search methods applied to parametric design of aircraft. *Proceedings Third International Conference on Genetic Algorithms,* 4-7 June 1989, George Mason University, 213-218.

Bratley, P., Fox, B.L. and Schrage, L.E. (1987). *A guide to simulation.* Springer-Verlag, New York.

Bunday, B.D. (1984). *Basic optimisation methods.* Edward Arnold, London.

Buxton, R. and Stafford-Smith, M. (1996). Managing drought in Australia's rangelands: four weddings and a funeral. *Rangelands Journal* **18,** 292-308.

Cacho, O. and Simmons, P. (1999). A genetic algorithm approach to farm investment. *Australian Journal of Agricultural and Resource Economics* **43,** 305-322.

Camerini, P.M., Fratta, L. and Maffioli, F. (1975). On improving relaxation methods by modified gradient techniques. In *Nondifferentiable optimization* (Eds, M.L. Balinski and P. Wolfe), North-Holland, Amsterdam.

Campbell, N.J., Hanrahan, P.D., Russell, I.M., Roberts, G.S. and Horton, B.J. (1998). Modelling pesticide residues on greasy wool: experimental studies. *Australian Journal of Experimental Agricultural* **38,** 441-449.

Cerny, V. (1985). A thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications* **45**, 41-51.

Cho, S.I. and Lee, J.H. (2000). Autonomous speedsprayer using differential global positioning system, genetic algorithm and fuzzy control. *Journal of Agricultural Engineering Research* **76,** 111-119.

Clark, S.G., Donnelly, J.R. and Moore, A.D. (2000). The GrassGro decision support tool: its effectiveness in simulating pasture and animal production and value in determining research priorities. *Australian Journal of Experimental Agricultural* **40,** 247-256.

Corana, A., Marchesi, M., Martini, C. and Ridella, S. (1987). Minimizing multimodal functions of continuous variables with the "simulated annealing" algorithm. *ACM Transactions on Mathematical Software* **13,** 262-280.

Cvijovic, D. and Klinowski, J. (1995). Taboo search: An approach to the multiple minima problem. *Science* **267,** 664-666.

Davidor, Y. (1991). A genetic algorithm applied to robot trajectory generation. In *Handbook of genetic algorithms* (Ed, L. Davis), Van Nostrand Reinhold, New York.

Davies, Z.S., Gilbert, R.J., Merry, R.J., Kell, D.B., Theodorou, M.K. and Griffith, G.W. (2000). Efficient improvement of silage additives by using genetic algorithms. *Applied and Environmental Microbiology* **66,** 1435-1443.

Davis, L. (1991a). Bit climbing representational bias and test suite design. In *Genetic algorithms* (Eds, R.K. Belew and L.B. Booker), Morgan Kaufmann, San Mateo, 18-23.

Davis, L. (1991b). *Handbook of genetic algorithms.* Van Nostrand Reinhold, New York.

Davis, L. and Steenstrup, M. (1987). Genetic algorithms and simulated annealing: An overview. In *Genetic algorithms and simulated annealing* (Ed, L. Davis), Pitman, London, 1-11.

Day, R.H. (1977). On economic optimization: A nontechnical survey. In *A survey of agricultural economics literature* (Eds, G.G. Judge, R.H. Day, S.R. Johnson, G.C. Rausser and L.R. Martin), University of Minnesota Press, Minneapolis.

Day, R.H. and Sparling, E. (1977). Optimization models in agricultural and resource economics. In *A survey of agricultural economics literature* (Eds, G.G. Judge, R.H. Day, S.R. Johnson, G.C. Rausser and L.R. Martin), University of Minnesota Press, Minneapolis.

De Jong, K. and Sarma, J. (1995). On decentralizing selection algorithms. *Proceedings Sixth International Conference on Genetic Algorithms,* 15-19 July 1995, University of Pittsburgh, 17-23.

de Juan, J.A., Tarjuelo, J.M., Ortega, J.F., Valiente, M. and Carriön, P. (1999). Management of water consumption in agriculture. A model for the economic optimisation of water use: application to a sub-humid area. *Agricultural Water Management* **40,** 303-313.

Dembo, R.S., Eisenstat, S.C. and Steihaug, T. (1982). Inexact Newton methods. *SIAM Journal of Numerical Analysis* **19,** 400-408.

Dennis, J.E. and More, J.J. (1977). Quasi-Newton methods motivation and theory. *SIAM Review* **19,** 46-89.

Dent, J.B. and Blackie, M.J. (1979). *Systems simulation in agriculture.* Applied Science Publishers, London.

Dunn, J.C. (1981). Global and asymptotic convergence rate estimates for a class of projected gradient processes. *SIAM Journal of Control and Optimization* **19,** 368-400.

Feinerman, E., Bresler, E. and Achrish, H. (1989). Economics of irrigation technology under conditions of spatially variable soils and nonuniform water distribution. *Agronomie* **9,** 819-826.

Feo, T.A. and Resende, M.G.C. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization* **6,** 109-133.

Ferreira, A.G. and Zerovnik, J. (1993). Bounding the probability of success of stochastic methods for global optimization. *Computers and Mathematics with Applications* **25,** 1-8.

Fisher, D.S. and Baumont, R. (1994). Modeling the rate and quantity of forage intake by ruminants during meals. *Agricultural Systems* **45,** 43-53.

Fisher, M.L., Northup, W.D. and Shapiro, J.F. (1975). Using duality to solve discrete optimization problems: Theory and computational experience. In *Nondifferentiable optimization* (Eds, M.L. Balinski and P. Wolfe), North-Holland, Amsterdam.

Fletcher, R. (1982). Methods for nonlinear constraints. In *Nonlinear optimisation 1981* (Ed, M.J.D. Powell), Academic, London.

Fletcher, R. (1987). *Practical methods of optimization.* Wiley, New York.

Fogel, D.B. (1995a). A comparison of evolutionary programming and genetic algorithms on selected constrained optimization problems. *Simulation* **64,** 399-406.

Fogel, D.B. (1995b). *Evolutionary computation - Toward a new philosophy of machine intelligence.* IEEE Press, New York.

Fogel, D.B. and Angeline, P.J. (1997). Guidelines for a suitable encoding. In *Handbook of evolutionary computation* (Eds, T. Bäck, D.B. Fogel and Z. Michalewicz), Institute of Physics Publishing, Bristol, C1.7:1-2.

Fonesca, C.M. and Fleming, P.J. (1997). Multobjective optimization. In *Handbook of evolutionary computation* (Eds, T. Bäck, D.B. Fogel and Z. Michalewicz), Institute of Physics Publishing, Bristol, C4.5:1-9.

Foran, B.D., Stafford-Smith, D.M., Niethe, G., Stockwell, T. and Michell, V. (1990). A comparison of development options on a northern Australian beef property. *Agricultural Systems* **34,** 77-102.

Fox, B.L. (1993). Integrating and accelerating tabu search, simulated annealing and genetic algorithms. *Annals of Operations Research* **41,** 47-67.

Franchini, M. (1996). Use of a genetic algorithm combined with a local search method for the automatic calibration of conceptual rainfall-runoff models. *Hydrological Sciences Journal* **41,** 21-39.

Franchini, M., Galeati, G. and Berra, S. (1998). Global optimization techniques for the calibration of conceptual rainfall-runoff models. *Hydrological Sciences Journal* **43,** 443-458.

Freer, M., Moore, A.D. and Donnelly, J.R. (1997). GRAZPLAN: Decision support systems for Australian grazing enterprises - II. The animal biology model for feed intake, production and reproduction and the GrazFeed DSS. *Agricultural Systems* **54,** 77-126.

Fryer, M.J. and Greenman, J.V. (1987). *Optimisation theory.* Edward Arnold, London.

Gabbert, P.S., Brown, D.E., Huntley, C.L., Markowicz, B.P. and Sappington, D.E. (1991). A system for learning routes and schedules with genetic algorithms. In *Genetic algorithms* (Eds, R.K. Belew and L.B. Booker), Morgan Kaufmann, San Mateo, 430-436.

Gates, R.H., Minagawa, H., Timmons, M.B. and Chi, H. (1994). Economic optimization of Japanese swine production. *International Summer Meeting of the American Society of Agricultural Engineers,* 19-22 June 1994, Kansas City, 1-15.

Gill, P.E., Murray, W. and Wright, M.H. (1981). *Practical optimisation.* Academic, London.

Gillard, P. and Monypenny, R. (1988). A decision support approach for the beef cattle industry of tropical Australia. *Agricultural Systems* **26,** 179-190.

Glass, C.A. and Potts, C.N. (1996). A comparison of local search methods for flow shop scheduling. *Annals of Operations Research* **63,** 489-509.

Gleick, J. (1987). *Chaos - Making a new science.* Penguin, London.

Glover, F. (1989). Tabu search - Part I. *ORSA Journal on Computing* **1,** 190-206.

Glover, F. (1990a). Tabu search - Part II. *ORSA Journal on Computing* **2,** 4-32.

Glover, F. (1990b). Tabu search: A tutorial. *Interfaces* **20,** 74-94.

Glover, F., Taillard, E. and de Werra, D. (1993). A user's guide to tabu search. *Annals of Operations Research* **41,** 3-28.

Goffe, W.L., Ferrier, G.D. and Rogers, J. (1994). Global optimization of statistical functions with simulated annealing. *Journal of Econometrics* **60,** 65-99.

Goggos, V. and King, R.E. (2000). Qualitative-evolutionary design of greenhouse environment control agents. *Computers and Electronics in Agriculture* **26,** 271-282.

Goldberg, D.E. (1987). Genetic algorithms with sharing for multimodal function optimization. In *Genetic algorithms and their applications* (Ed, J.J. Grefenstette), Lawrence Erlbaum, Hillsdale, 41-49.

Goldberg, D.E. (1989). *Genetic algorithms in search, optimization and machine learning.* Addison-Wesley, Reading.

Goldberg, D.E., Deb, K. and Clark, J.H. (1992). Genetic algorithms, noise, and the sizing of populations. *Complex Systems* **6,** 333-362.

Grebogi, C., Ott, E. and Yorke, J.A. (1987). Chaos, strange attractors and fractal basin boundaries in nonlinear dynamics. *Science* **238,** 632-638.

Grefenstette, J.J. (1995). A user's guide to the genetic search implementation system GENESIS 5.0. http://www.aic.nrl.mil/

Griffith, G.R. and Piggott, N.E. (1994). Asymmetry in beef, lamb and pork farm-retail price transmission in Australia. *Agricultural Economics* **10,** 307-316.

Hall, W.B., McKeon, G.M., Carter, J.O., Day, K.A., Howden, S.M., Scanlan, J.C., Johnston, P.W. and Burrows, W.H. (1998). Climate change in Queensland's grazing lands: II. An assessment of the impact on animal production from native pastures. *Rangelands Journal* **20,** 177-205.

Hammel, U. (1997). Simulation models. In *Handbook of evolutionary computation* (Eds, T. Bäck, D.B. Fogel and Z. Michalewicz), Institute of Physics Publishing, Bristol, F1.8:1-9.

Hammer, G. and White, D. (1992). From reality to research and back again by computer model. *Agricultural Science* **5,** 35-39.

Hammer, G.L., Carberry, P.S. and Muchow, R.C. (1993). Modelling genotypic and environmental control of leaf area dynamics in grain sorghum. I. Whole plant level. *Field Crops Research* **33,** 293-310.

Hancock, P.J.B. (1997). A comparison of selection mechanisms. In *Handbook of evolutionary computation* (Eds, T. Bäck, D.B. Fogel and Z. Michalewicz), Institute of Physics Publishing, Bristol, C2.8:1-11.

Hansen, J.W. and Jones, J.W. (2000). Scaling-up models for climatic variability applications. *Agricultural Systems* **65,** 43-72.

Hansen, P. and Jaumard, B. (1990). Algorithms for the maximum satisfiability problem. *Computing* **44,** 279-303.

Hart, R.P.S., Larcombe, M.T., Sherlock, R.A. and Smith, L.A. (1998). Optimisation techniques for a computer simulation of a pastoral dairy farm. *Computers and Electronics in Agriculture* **19,** 129-153.

Hart, W.E. (1996). A theoretical comparison of evolutionary algorithms and simulated annealing. *Proceedings Fifth Annual Conference on Evolutionary Programming,* 29 February - 2 March 1996, San Diego, California

Hayes, B.J., Shepherd, R.K. and Newman, S. (1997). Selecting mating pairs with genetic algorithms. *Proceedings 12th Conference, Association for the Advancement of Animal Breeding and Genetics,* 6-10 April 1997, Dubbo, 108-112.

Hendrickson, J.D., Sorooshian, S. and Brazil, L.E. (1988). Comparison of Newton-type and direct search algorithms for calibration of conceptual rainfall-runoff models. *Water Resources Research* **24,** 691-700.

Hesser, J., Manner, R. and Stucky, O. (1989). Optimization of Steiner trees using genetic algorithms. *Proceedings Third International Conference on Genetic Algorithms,* 4-7 June 1989, George Mason University, 231-236.

Hill, M.J., Donald, G.E., Vickery, P.J. and Furnival, E.P. (1996). Integration of satellite remote sensing, simple bioclimatic models and GIS for assessment of pastoral development for a commercial grazing enterprise. *Australian Journal of Experimental Agriculture* **36,** 309-321.

Hinterding, R., Gielewski, H. and Peachey, T.C. (1995). The nature of mutation in genetic algorithms. *Proceedings Sixth International Conference on Genetic Algorithms,* 15-19 July 1995, University of Pittsburgh, 65-72.

Holak, T.A., Gondal, D., Otlewski, J. and Wilusz, T. (1989). Determination of the complete three-dimensional structure of the trypsin inhibitor from squash seeds in aqueous solution by nuclear magnetic resonance and a combination of distance geometry and dynamical simulated annealing. *Journal of Molecular Biology* **210,** 635-648.

Honghai, D. (1987). Ecological-economic systems analysis of agricultural structure. *Resource Management and Optimization* **4,** 141-166.

Horton, B. (1996). A method of using a genetic algorithm to examine the optimum structure of the Australian sheep breeding industry: Open-breeding systems MOET and AI. *Australian Journal of Experimental Agriculture* **36,** 249-258.

Houck, C.R., Joines, J.A. and Kay, M.G. (1996). Comparison of genetic algorithms, random restart, and two-opt switching for solving large location-allocation problems. *Computers and Operations Research* **23,** 587-596.

Howden, S.M., Moore, J.L., McKeon, G.M., Reyenga, P.J., Carter, J.O. and Scanlan, J.C. (1999). Dynamics of mulga woodlands in south-west Queensland: global change impacts and adaptation. *Proceedings International Congress on Modelling and Simulation,* December 1999, The University of Waikato, Hamilton, 637-642.

Husmann, H.-J. and Tantau, H.-J. (2001). Integrated optimization of energy supply systems in horticulture using genetic algorithms. *Computers and Electronics in Agriculture* **31,** 47-59.

Ibarki, T. (1997a). Comparison with existing optimization methods. In *Handbook of evolutionary computation* (Eds, T. Bäck, D.B. Fogel and Z. Michalewicz), Institute of Physics Publishing, Bristol, D3.6:1-2.

Ibarki, T. (1997b). Simulated annealing and tabu search. In *Handbook of evolutionary computation* (Eds, T. Bäck, D.B. Fogel and Z. Michalewicz), Institute of Physics Publishing, Bristol, D3.5:1-2.

Ingber, L. (1989). Very fast simulated re-annealing. *Mathematical and Computer Modelling* **12,** 967-973.

Ingber, L. (1993). Simulated annealing: Practice versus theory. *Mathematical and Computer Modelling* **18,** 29-57.

Ingber, L. (1996). Adaptive simulated annealing ASA: Lessons learned. *Control and Cybernetics* **25,** 33-54.

Ingber, L. and Rosen, B. (1992). Genetic algorithms and very fast simulated reannealing: A comparison. *Mathematical and Computer Modelling* **16,** 87-100.

Jones, T. (1995). Crossover macromutation and population-based search. *Proceedings Sixth International Conference on Genetic Algorithms,* 15-19 July 1995, University of Pittsburgh, 73-80.

Judge, G.G., Day, R.H., Johnson, S.R., Rausser, G.C. and Martin, L.R. (Eds.) (1977). *A survey of agricultural economics literature.* University of Minnesota Press, Minneapolis.

Karr, C.L. (1991). Air-injected hydrocyclone optimization via genetic algorithm. In *Handbook of genetic algorithms* (Ed, L. Davis), Van Nostrand Reinhold, New York.

Keane, A.J. (1996). A brief comparison of some evolutionary optimization methods. In *Modern heuristic search methods* (Eds, V. Raywood-Smith, I. Osman, C. Reeves and G.D. Smith), Wiley, New York, 255-272.

King, G.A. and Logan, S.H. (1964). Optimum location number and size on processing plants with raw product and final product shipments. *Journal of Farm Economy* **46,** 94-108.

Kirkpatrick, S., Gelatt, C.D. and Vecchi, M.P. (1983). Optimization by simulated annealing. *Science* **220,** 671-680.

Kleijnen, J.P.C. (1987). *Statistical tools for simulation practitioners.* Marcel Dekker, New York.

Klimasauskas, C.C. (1992). Genetic function optimization for time series prediction. *Advanced Technology for Developers* **1,** 5-9.

Koo, W.W. and Thompson, S. (1982). An economic analysis of U.S. grain marketing and transport system. Research Report 89, North Dakota State University, Fargo.

Koza, J.R. (1994). *Genetic programming II. Automatic discovery of reusable programs.* MIT Press, Cambridge MA.

Kozan, E. (1999). Heuristic techniques to increase the efficiency of dispatching process in nurseries. *Proceedings International Congress on Modelling and Simulation,* December 1999, The University of Waikato, Hamilton, 913-918.

Kulshreshtha, S.N., Junkins, B. and Desjardins, R. (2000). Prioritizing greenhouse gas emission mitigation measures for agriculture. *Agricultural Systems* **66,** 145-166.

Kuo, C.-H., Michel, A.N. and Gray, W.G. (1992). Design of optimal pump-and-treat strategies for contaminated groundwater remediation using the simulated annealing algorithm. *Advances in Water Resources* **15,** 95-105.

Kuo, S.-F., Merkley, G.P. and Liu, C.-W. (2000). Decision support for irrigation project planning using a genetic algorithm. *Agricultural Water Management* **45,** 243-266.

Lagarias, J.C., Reeds, J.A., Wright, M.H. and Wright, P.E. (1998). Convergence properties of the Nelder-Mead simplex algorithm in low dimensions. *SIAM Journal on Optimization* **9,** 112-147.

Lambert, D.K. and Harris, T.R. (1990). Stochastic dynamic optimization and rangeland investment decisions. *Western Journal of Agricultural Economics* **15,** 186-195.

Law, A.M. and Kelton, W.D. (1982). *Simulation modelling and analysis.* McGraw-Hill, New York.

Lazarus, W.F. and Dixon, B.L. (1984). Agricultural pests as common property: Control of the corn rootworm. *American Journal of Agricultural Economics* **66,** 456-465.

Le Riche, R.G., Knopf-Lenoir, C. and Haftka, R.T. (1995). A segregated genetic algorithm for constrained structural optimization. *Proceedings Sixth International Conference on Genetic Algorithms,* 15-19 July 1995, University of Pittsburgh, 558-565.

Liedloff, A.C., Coughenour, M.B., Ludwig, J.A. and Dyer, R. (1999). Modelling the trade-off between fire and grazing in a tropical Savanna landscape, northern Australia. *Proceedings International Congress on Modelling and Simulation,* December 1999, The University of Waikato, Hamilton, 673-678.

Lockwood, C. and Moore, T. (1993). Harvest scheduling with spatial constraints : A simulated annealing approach. *Canadian Journal of Forestry Research* **23,** 468-478.

Lu, F. and Eriksson, L.O. (2000). Formation of harvest units with genetic algorithms. *Forest Ecology and Management* **130,** 57-67.

Lundy, M. (1985). Applications of the annealing algorithm to combinatorial problems in statistics. *Biometrika* **72,** 191-198.

MacKinnon, J.G. (1976). A technique for the solution of spatial equilibrium models. *Journal of Regional Science* **16,** 293-307.

Madsen, K. (1975). Minimax solution of non-linear equations without calculating derivatives. In *Nondifferentiable optimization* (Eds, M.L. Balinski and P. Wolfe), North-Holland, Amsterdam.

Mallawaarachchi, T., Walker, P.A., Young, M.D., Smyth, R.E. and Lynch, H.S. (1996). GIS-based integrated modelling systems for natural resource management. *Agricultural Systems* **50,** 169-189.

Mardle, S. and Pascoe, S. (2000). Use of evolutionary methods for bioeconomic optimization models: an application to fisheries. *Agricultural Systems* **66,** 33-49.

Mariano, C.E. (1998). Optimal design of water distribution networks using genetic algorithms. *Proceedings Seventh International Conference on Computers in Agriculture,* 26-30 October 1998, Hyatt Orlando, 146-158.

Mayer, D.G. (1985). Simulation of small systems using microcomputers. *Mathematics and Computers in Simulation* **27,** 207-210.

Mayer, D.G. (1989). Fertilizer optimisation package. *Proceedings Fourth National Computers in Agriculture Conference,* 8-12 May 1989, Maroochydore, Australia, 63.

Mayer, D.G. (2000). *Optimisation of agricultural production systems models.* Ph.D. Thesis, University of Queensland, Brisbane.

Mayer, D.G., Belward, J.A. and Burrage, K. (1996a). Use of advanced techniques to optimize a multi-dimensional dairy model. *Agricultural Systems* **50,** 239-253.

Mayer, D.G., Belward, J.A. and Burrage, K. (1997). Use of evolutionary algorithms and simulated annealing to optimize a herd dynamics model. *Proceedings International Congress on Modelling and Simulation,* 8-11 December 1997, University of Tasmania, Hobart, 1109-1114.

Mayer, D.G., Belward, J.A. and Burrage, K. (1998a). Optimizing simulation models of agricultural systems. *Annals of Operations Research* **82,** 219-231.

Mayer, D.G., Belward, J.A. and Burrage, K. (1998b). Tabu search not an optimal choice for models of agricultural systems. *Agricultural Systems* **58,** 243-251.

Mayer, D.G., Belward, J.A. and Burrage, K. (1999a). Performance of genetic algorithms and simulated annealing in the economic optimization of a herd dynamics model. *Environment International* **25,** 899-905.

Mayer, D.G., Belward, J.A. and Burrage, K. (2001). Robust parameter settings of evolutionary algorithms for the optimisation of agricultural systems models. *Agricultural Systems* **69,** 199-213.

Mayer, D.G., Belward, J.A., Burrage, K. and Stuart, M.A. (1995). Optimisation of a dairy farm model - Comparison of simulated annealing, simulated quenching and genetic algorithms. *Proceedings 1995 International Congress on Modelling and Simulation,* 27-30 November 1995, University of Newcastle, Newcastle, 33-38.

Mayer, D.G., Belward, J.A., Widell, H. and Burrage, K. (1999b). Survival of the fittest - Genetic algorithms versus evolution strategies in the optimization of systems models. *Agricultural Systems* **60,** 113-122.

Mayer, D.G. and Butler, D.G. (1993). Statistical validation. *Ecological Modelling* **68,** 21-32.

Mayer, D.G., Kelly, A.M. and Butler, D.G. (1989). A review of optimisation techniques for non-linear simulation models. *Proceedings 8th Biennial Conference on Modelling and Simulation,* 25-27 September 1989, Australian National University, Canberra, 229-234.

Mayer, D.G., Pepper, P.M., Belward, J.A., Burrage, K. and Swain, A.J. (1996b). Simulated annealing - A robust optimization technique for fitting nonlinear regression models. *Proceedings 'Modelling Simulation and Optimization' Conference, International Association of Science and Technology for Development,* 6-9 May 1996, Gold Coast, Australia

Mayer, D.G., Pepper, P.M., McKeon, G.M. and Moore, A.D. (1999c). Modelling mortality and reproduction rates for management of beef herds in northern Australia. *Proceedings 'Applied Modelling and Simulation' Conference, International Association of Science and Technology for Development,* Acta Press, 1-3 September 1999, Cairns, 142-147.

Mayer, D.G., Schoorl, D., Butler, D.G. and Kelly, A.M. (1991). Efficiency and fractal behaviour of optimisation methods on multiple-optima surfaces. *Agricultural Systems* **36,** 315-328.

Mayer, D.G., Stuart, M.A. and Swain, A.J. (1994a). Regression of real-world data on model output: An appropriate overall test of validity. *Agricultural Systems* **45,** 93-104.

Mayer, D.G., Tierney, M.L. and Thurbon, P.N. (1994b). Statistical experiments with simulation models - A dairy genetics example. *Agricultural Systems* **45,** 203-216.

McIvor, J.G. and Monypenny, R. (1995). Evaluation of pasture management systems for beef production in the semi-arid tropics: Model development. *Agricultural Systems* **49,** 45-67.

McKeon, G.M., Day, K.A., Howden, S.M., Mott, J.J., Orr, D.M., Scattini, W.J. and Weston, E.J. (1990). Northern Australian savannas: Management for pastoral production. *Journal of Biogeography* **17,** 355-372.

Meadows, D.H. and Robinson, J.M. (1985). *The electronic oracle.* Wiley, New York.

Messine, F., Lagouanelle, J.L., Noailles, J., Pibouleau, L., Domenech, S. and Floquet, P. (1996). Global optimization of chemical engineering processes. *Proceedings 'Modelling*

Simulation and Optimization Conference, International Association of Science and Technology for Development, 6-9 May 1996, Gold Coast, Australia

Meszaros, S.A., Banks, R.G., van der Werf, J.H.J. and Goddard, M. (1999). The use of genetic algorithms for optimizing age structure in breeding populations when inbreeding depresses genetic gain through effects on reproduction. *Animal Science* **68,** 449-457.

Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H. and Teller, E. (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics* **21,** 1087-1092.

Michalewicz, Z. (1996). *Genetic algorithms + data structures = evolution programs.* Springer, Berlin.

Michalewicz, Z. and Fogel, D.B. (2000). *How to solve it: Modern heuristics.* Springer, Berlin.

Moore, C.T., Conroy, M.J. and Boston, K. (2000). Forest management decisions for wildlife objectives: system resolution and optimality. *Computers and Electronics in Agriculture* **27,** 25-39.

Morimoto, T. and Hashimoto, Y. (1996). Optimal control of plant growth in hydroponics using neural networks and genetic algorithms. *Acta Horticulturae* **406,** 433-440.

Morris, M.D., Mitchell, T.J. and Ylvisaker, D. (1993). Bayesian design and analysis of computer experiments: Use of derivatives in surface prediction. *Technometrics* **35,** 243-255.

Mühlenbein, H. and Schlierkamp-Voosen, D. (1993). Predictive models for the breeder genetic algorithm. *Evolutionary Computation* **1,** 25-49.

Mühlenbein, H. and Schlierkamp-Voosen, D. (1994). The science of breeding and its application to the breeder genetic algorithm BGA. *Evolutionary Computation* **1,** 335-360.

Murtagh, F. (1994). Neural networks and related 'massively parallel' methods for statistics: a short overview. *International Statistical Review* **62,** 275-288.

Nelder, J.A. and Mead, R. (1965). A simplex method for function minimisation. *The Computer Journal* **7,** 308-313.

Notter, D.R. and Johnson, M.H. (1987). Simulation of genetic control of reproduction in beef cows. III. Within-herd breeding value estimation with known breeding dates. *Journal of Animal Science* **65,** 88-98.

Notter, D.R. and Johnson, M.H. (1988). Simulation of genetic control of reproduction in beef cows. IV. Within-herd breeding value estimation with pasture mating. *Journal of Animal Science* **66,** 280-286.

Olsen, J.K., McMahon, C.R. and Hammer, G.L. (1993). Prediction of sweet corn phenology in subtropical environments. *Agronomy Journal* **85,** 410-415.

Onstad, D.W. (1988). Population-dynamics theory: the roles of analytical, simulation, and supercomputer models. *Ecological Modelling* **43,** 111-124.

O'Rourke, P.K., Winks, L. and Kelly, A.M. (1992). *North Australia beef producer survey 1990.* Queensland Department of Primary Industries, Brisbane.

Osman, I.H. (1993). Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Annals of Operations Research* **41,** 421-451.

Pabico, J.P., Hoogenboom, G. and McClendon, R.W. (1999). Determination of cultivar coefficients of crop models using a genetic algorithm: a conceptual framework. *Transactions of the ASAE* **42,** 223-232.

Pardalos, P.M., Pitsoulis, L., Mavridou, T. and Resende, M.G.C. (1995). Parallel search for combinatorial optimization: Genetic algorithms, simulated annealing, tabu search and GRASP. *Lecture Notes in Computer Science* **980,** 317-331.

Park, K. (1995). A comparative study of genetic search. *Proceedings Sixth International Conference on Genetic Algorithms,* 15-19 July 1995, University of Pittsburgh, 512-519.

Parker, W.J., Dooley, A.E. and Dake, C.K. (1999). Decision support for farm business strategy: An example for sheep breeding. *Proceedings International Congress on Modelling and Simulation,* December 1999, The University of Waikato, Hamilton, 1021-1026.

Parmar, R.S., McClendon, R.W. and Potter, W.D. (1996). Farm machinery selection using simulation and genetic algorithms. *Transactions of the ASAE* **39,** 1905-1909.

Parsons, D.J. (1998). Optimising silage harvesting plans in a grass and grazing simulation using the revised simplex method and a genetic algorithm. *Agricultural Systems* **56,** 29-44.

Paudel, K.P., Lohr, L. and Martin, N.R.J. (2000). Effect of risk perspective on fertilizer choice by sharecroppers. *Agricultural Systems* **66,** 115-128.

Paz, J.O., Batchelor, W.D., Babcock, B.A., Colvin, T.S., Logsdon, S.D., Kaspar, T.C. and Karlen, D.L. (1999). Model-based technique to determine variable rate nitrogen for corn. *Agricultural Systems* **61,** 69-75.

Peck, C.C. and Dhawan, A.P. (1995). Genetic algorithms as global random search methods: An alternative perspective. *Evolutionary Computation* **3,** 39-80.

Pelikan, M. and Goldberg, D.E. (2001). Escaping heirarchical traps with competent genetic algorithms. IlliGAL Report No. 2001003, University of Illinois, Urbana-Champaign.

Pelikan, M., Goldberg, D.E. and Cantú-Paz, E. (2000). Bayesian optimization algorithm, population sizing, and time to convergence. IlliGAL Report No. 2000001, University of Illinois, Urbana-Champaign.

Peterson, C. and Söderberg, B. (1989). A new method for mapping optimization problems onto neural networks. *International Journal of Neural Systems* **1,** 3-22.

Pluimers, J.C., Kroeze, C., Bakker, E.J., Challa, H. and Hordijk, L. (2000). Quantifying the environmental impact of production in agriculture and horticulture in The Netherlands: which emissions do we need to consider? *Agricultural Systems* **66,** 167-189.

Polheim, H. and Heißner, A. (1997). Optimal control of greenhouse climate using evolutionary algorithms. *Proceedings International Scientific Colloquium IWK'97,* 22-25 September 1997, Ilmenau, Germany, 1-8.

Polyak, B.T. (1969). Minimization of unsmooth functionals. *USSR Computational Mathematics and Mathematical Physics* **9,** 14-29.

Polyak, B.T. (1987). *Introduction to optimization.* Optimization Software, New York.

Powell, M.J.D. (1964). An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal* **7,** 155-162.

Powell, M.J.D. (1982). *Nonlinear optimization 1981.* Academic, London.

Pratt, J.E., Novakovic, A.M., Elterich, G.J., Hahn, D.E., Smith, B.J. and Criner, G.K. (1986). An analysis of the spatial organization of the northeast dairy industry. *Search: Agriculture* **32,** 1-35.

Rabow, A.A. and Scheraga, H.A. (1993). Lattice neural network minimization. Application of neural network optimization for locating the global-minimum conformations of proteins. *Journal of Molecular Biology* **232,** 1157-1168.

Radcliffe, N. and Wilson, G. (1990). Natural solutions give their best. *New Scientist* **126,** 35-38.

Richardson, F.D. and Hahn, B.D. (1994). Models for the selection of cow types for extensive meat and milk production in developing areas. Network Paper 36d, Overseas Development Institute, London.

Rickert, K.G. and Winter, W.H. (1980). Integration of feed sources in property management: Extensive systems. *Tropical Grasslands* **14,** 239-245.

Ripley, B.D. (1987). *Stochastic simulation.* Wiley, New York.

Ritter, H., Martinetz, T. and Schulten, K. (1992). *Neural computation and self-organising maps.* Addison-Wesley, Reading.

Ritzel, B.J. and Eheart, J.W. (1994). Using genetic algorithms to solve a multiple objective groundwater pollution containment problem. *Water Resources Research* **30,** 1589-1603.

Rodriguez, A., Trapp, J.N., Walker, O.L. and Bernardo, D.J. (1990). A wheat grazing systems model for the US southern plains: Part II - Economic analysis of grazing management decisions under weather uncertainty. *Agricultural Systems* **33,** 61-75.

Roise, J.P. (1990). Multicriteria nonlinear programming for optimal spatial allocation of stands. *Forest Science* **36,** 487-501.

Rossing, W.A.H., van der Werf, W. and Rabbinge, R. (1997). Systems research in support of crop and plant health: The role of production ecology. *Agricultural Systems & Information Technology* **71,** 5-12.

Rothlauf, F., Goldberg, D.E. and Heinzl, A. (2000). Bad codings and the utility of well-designed genetic algorithms. IlliGAL Report No. 2000007, University of Illinois, Urbana-Champaign.

Saloman, R. (1996). Re-evaluating genetic algorithm performance under coordinate rotation of benchmark functions. A survey of some theoretical and practical aspects of genetic algorithms. *BioSystems* **39,** 263-278.

Schaffer, J.D., Caruana, R.A., Eshelman, L.J. and Das, R. (1989). A study of control parameters affecting online performance of genetic algorithms for function optimization. *Proceedings Third International Conference on Genetic Algorithms,* 4-7 June 1989, George Mason University, 51-60.

Schilizzi, S.G.M. and Kingwell, R.S. (1999). Effects of climate and price uncertainty on the value of legume crops in a Mediterranean-type environment. *Agricultural Systems* **60,** 55-69.

Schwefel, H.-P. (1997). Challenges to and future developments of evolutionary algorithms. In *Handbook of evolutionary computation* (Eds, T. Bäck, D.B. Fogel and Z. Michalewicz), Institute of Physics Publishing, Bristol, H1.3:1-4.

Shaw, I. and Findlay, C.J. (1990). Forecasting Victorian fleece weights using a climatic pasture index. *Agricultural Systems* **34,** 191-205.

Shor, N.Z. (1985). *Minimization methods for non-differentiable functions.* Springer-Verlag, Berlin.

Smale, S. (1986). Algorithms for solving equations. *Proceedings of the International Congress of Mathematicians,* University of California, Berkeley, 172-195.

Smith, R.E. (1997). Population size. In *Handbook of evolutionary computation* (Eds, T. Bäck, D.B. Fogel and Z. Michalewicz), Institute of Physics Publishing, Bristol, E1.1:1-5.

South, M.C., Wetherill, G.B. and Tham, M.T. (1993). Hitch-hiker's guide to genetic algorithms. *Journal of Applied Statistics* **20,** 153-175.

Späth, E.J.A., Morley, F.H.W. and White, D.H. (1984). A computer model of a self-replacing beef herd. *Proceedings Australian Society of Animal Production* **15,** 597-600.

Spears, W.M. (1997). Recombination parameters. In *Handbook of evolutionary computation* (Eds, T. Bäck, D.B. Fogel and Z. Michalewicz), Institute of Physics Publishing, Bristol, E1.3:1-11.

Spears, W.M. (2000). *Evolutionary algorithms. The role of mutation and recombination.* Springer, Berlin.

Spendley, W., Hext, G.R. and Himsworth, F.R. (1962). Sequential application of simplex designs in optimisation and evolutionary operation. *Technometrics* **4,** 441-461.

Stafford-Smith, D.M. and McKeon, G.M. (1998). Assessing the historical frequency of drought events on grazing properties in Australian rangelands. *Agricultural Systems* **57,** 271-299.

Stevens, B.H. (1968). Location theory and programming models: The von Thünen case. *Papers of the Regional Science Association* **21,** 19-34.

Styblinski, M.A. and Tang, T.-S. (1990). Experiments in nonconvex optimization: Stochastic approximation with function smoothing and simulated annealing. *Neural Networks* **3,** 467-483.

Syswerda, G. (1991). Schedule optimization using genetic algorithms. In *Handbook of genetic algorithms* (Ed, L. Davis), Van Nostrand Reinhold, New York.

Szu, H. and Hartley, R. (1987). Fast simulated annealing. *Physics Letters A* **122,** 157-162.

Thornton, P.K. (1988). An animal production model for assessing the bio-economic feasibility of various management strategies for the isohyperthermic savannas of Colombia. *Agricultural Systems* **27,** 137-156.

Thornton, P.K. and McGregor, M.J. (1988). The identification of optimum management regimes for agricultural crop enterprises. *Outlook on Agriculture* **17,** 158-162.

van Laarhoven, P.J.M. and Aarts, E.H.L. (1987). *Simulated annealing: Theory and applications.* Reidel, Dordrecht.

Varanelli, J.M. and Cohoon, J.P. (1995). Population-oriented simulated annealing: A genetic/thermodynamic hybrid approach to optimization. *Proceedings Sixth*

International Conference on Genetic Algorithms, 15-19 July 1995, University of Pittsburgh, 174-181.

Verryn, S.D. and Roux, C.Z. (1998). The use of a simple genetic algorithm as a tree breeding selection technique. *Southern African Forestry Journal* **182,** 1-4.

Wagner, B.J. (1995). Sampling design methods for groundwater modeling under uncertainty. *Water Resources Research* **31,** 2581-2591.

Wang, Q.J. (1991). The genetic algorithm and its application to calibrating conceptual rainfall-runoff models. *Water Resources Research* **27,** 2467-2471.

Watson, R.A. and Sumner, N.R. (1999). Uncertainty and risk associated with optimised fishing patterns in a tropical penaeid fishery. *Environment International* **25,** 735-744.

Wayner, P. (1991). Genetic algorithms. *Byte*, January 1991, 361-368.

Whigham, P.A. (1999). Predictive modelling of plankton dynamics in freshwater lakes using genetic programming. *Proceedings International Congress on Modelling and Simulation,* December 1999, The University of Waikato, Hamilton, 679-684.

White, D.H., Howden, S.M. and Nix, H.A. (1996). Modelling agricultural and pastoral systems under environmental change. *Ecological Modelling* **86,** 213-217.

White, D.H., Howden, S.M., Walcott, J.J. and Cannon, R.M. (1998). A framework for estimating the extent and severity of drought based on a grazing system in south-eastern Australia. *Agricultural Systems* **57,** 259-270.

Widell, H. (1997). GENIAL II. A function optimizer based on evolutionary algorithms 1.1. http://www.hjem.get2net.dk/widell/genial.htm

Wolfe, P. (1975). A method of conjugate subgradients. In *Nondifferentiable optimization* (Eds, M.L. Balinski and P. Wolfe), North-Holland, Amsterdam.

Wolpert, D.H. and Macready, W.G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation* **1,** 67-82.

Woodward, S.J.R. (1998). Dynamical systems models and their application to optimizing grazing management. In *Agricultural systems modeling and simulation* (Eds, R.M. Peart and R.B. Curry), Marcel Dekker, New York, 419-473.

Wu, K. and Levine, M.D. (1994). Recovering parametric geons from multiview range data. *Proceedings IEEE Conference on Computer Vision and Pattern Recognition,* Seattle, 159-166.

Index