

https://dspace.org/

Dynamic simulation modelling of locust outbreaks

Tilch, David

1998
https://hdl.handle.net/10539/27348

Downloaded from a DSpace repository. https://dspace.org/



DYNAMIC SIMULATION MODELLING OF LOCUST OUTBREAKS

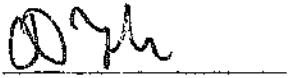
Submitted by: David Tilch

Dissertation submission to the Faculty of Science, University of the Witwatersrand, in fulfillment of the requirements for Masters of Science.

Johannesburg · 1998

■ DECLARATION ■

I declare that this dissertation is my own, unaided work. It is submitted for the degree of Masters of Science, in the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination in any other university.



David Tilch

Wednesday, April 1, 1998

Dynamic Simulation Modelling of Locust Outbreaks

Abstract

Much about the life history and biology of the Brown Locust, *Locustana pardalina (Walker)*, is known and documented; little is known of how locust populations change and move, owing to their occurrence in sparsely inhabited regions of southern Africa.

With a view to understanding the mechanisms driving population fluctuations, a model for simulating plague events was developed for the Mathworks Matlab™ interpreter. A previously developed model, applying rules that determine the transitions between the stages of the locust life-cycle, contingent on daily rainfall and temperature records, was enhanced by migration probability functions based on the distance between stations, wind speed, locust flight duration and wind heading.

It was hypothesised that incorporating the phenomenon of wind-facilitated migration into the basic model would increase the accuracy of predictions. The inclusion of migration tracking routines manifests itself in improvements in the simulation of outbreak events with respect to timing.

Acknowledgements

Thanks to the following, who partially funded the locust EWS project:

The Directorate of Resource Conservation, National Department of Agriculture

Thanks to the following organisations - in no particular order - who provided information and data essential to the development of the simulation model:

The Institute for Soil, Climate and Water, Pretoria, and *The Agricultural Research Council, Pretoria*	for co-ordinating the locust early warning system project and providing model test data, both locust control data and meteorological data.
The South African Weather Bureau, Pretoria	for providing meteorological data and access to simulated wind data records from the ETA model.
Computing Centre for Water Research, University of Natal	for providing access to their meteorology database.
The Plant Protection Institute, Pretoria	for providing locust control data and for providing biological information on the Brown Locust and its behaviour.
National Weather Service of the USA	for allowing the use of the ETA predictive wind simulation results as test data.

And to the following people for their contributions:

Dr. Dick Brown, of the P.P.R.I.	for the use of his wind-based swarm-movement maps.
Mr Neil Caithness, formerly of WITS Zoology	for assistance and tutorials on the use and operation of Matlab™.
Prof. Shirley Hanrahan, of WITS Zoology	for taking on the rôle of MSc supervisor, advising on, and guiding the development of the Simulation Model; for liaising with other organisations involved, notably the ARC and the PPRI; for providing information concerning the biological aspects of the Brown Locust lifecycle and behaviour, and for providing locust control data in spread-sheet format.
Mr Paul Hanrahan	for information and tutorials on the operation of early editions of locust lifecycle models, and for making the source code of those models available.
Ms. Margaret Kelser, of the P.P.R.I.	for computer processing of the locust control data files and making them available in D-Base™ format.
Mr Paul Nalland, formerly of WITS Zoology	for access to his early Pascal models of locust lifecycle simulations.
Mr. Peter Napier-Backs, of the P.P.R.I.	for biological information on Brown Locust swarming and behaviour, and providing records of recent swarm activities.

Mr. Eugene Poolman, of the S.A.W.B. for information on the ETA predictive wind simulation model, and access to simulated wind data.

Mr. Roger Price, of the P.P.R.I. for information on the lifecycle, swarming and migration behaviour of the Brown Locust.

Ms. Jane Swanepoel, of the C.C.W.R. for providing access to the CCWR's meteorological database.

Mev. Swart of the S.A.W.B. for providing extensive meteorological data on floppy disk.

Mrs. Anneke Thackra of the I.S.C.W. for co-ordinating the Locust Early Warning System project, liaising with the various organisations involved, and for providing locust control data and meteorological data.

Mej. Christa Van Heerden, formerly of the I.S.C.W. for computer processing of the locust control data files and making them available in D-Base™ format.

Table of contents

Title page
Declaration i
Abstract ii
Acknowledgements iii
Table of contents v
List of figures ix
List of tables xi
List of symbols xii

Chapter 1 Outline and objectives

1.1 Towards developing a locust outbreak early warning system 1
Potential impact of this research 2
Dynamic Simulation Modelling 3
1.2 The scope of this dissertation and work accomplished 4

Chapter 2 Biological and agricultural background

2.1 The need for a locust outbreak simulation model 6
A basic population model 8
2.2 Atmospheric phenomena and insect trajectories 9
Taking population migrations into account 9
Incorporating wind effects into the model 12
Migration sources and sinks 14
Practical modelling considerations 16
2.3 Related model building efforts 17
Wind transport modelling 18
Direction of migratory movement and insect orientation 20
Other factors influencing population redistribution 26
Integrating observed patterns into a simulation model 27

Chapter 3 The modelling approach

3.1 Mathematical models described 28
Stochastic and deterministic models 30
3.2 A history of the simulation model's development 31
A first approach to developing a model 31
Developments within the context of the dynamic simulation model 34
The lifecycle model 35
Object orientated programming systems (OOPS) 36
Black boxes and interfaces and ports 36
The locust lifecycle 37
How the lifecycle simulation operates I : loop stability 39
How the lifecycle simulation operates II : data processing 41
State variables 42
Chaotic behaviour described 43
Applying the rulebase 44
Initial conditions 45
3.3 Incorporation of migratory phenomena 46
The effect of wind on the migration path 47
Wind data processing-probabilities based on wind speed 49
Calculating the P values for wind speeds 52
Immigrants and emigrants; the source and destination sites 53
Wind direction probability functions 54
Calculation of P functions for wind headings 55
The von Mises circular probability distribution 57

Combining the probability functions 59
Migration and non-migration models 60
3.4 **Limitations on model complexity** 61
Computer hardware considerations 61
Determnining the quality of the model 63

Chapter 4 Model testing and evaluation
4.1 **Model testing concepts** 64
Verification, validation and sensitivity 64
4.2 **Model testing criteria** 65
Requirements for test simulations 65
Selection of test sites 67
Emulating soil temperatures 69
Reliability of data for chosen sites 73
Temperature and rainfall measurement accuracy 75
Site permutations 75
Initial conditions and outbreaks in 1985 76
Start and end dates: general test simulations 79
4.3 **Simulation run output processing** 79
Model test conditions 80
4.4 **A statistical approach** 81
Random variables 81
Joint probability functions 82
Tests of the distribution-free type 84
Spearman's rank correlation 85
Kendall's rank correlation 87
Interpretation of correlation coefficients 87
Dealing with tied data 88
Two tests based on the cumulative distribution function 89
Mann-Whitney tests 89
The Kolmogorov-Smirnov test 90
A simple non-statistcal evaluation 92
Summary 93

Chapter 5 Comments on test simulation runs
5.1 **Observations on the simulation output** 94
5.2 **Statistical analysis of the simulation results** 96
5.3 **A non-statistical analysis** 98
5.4 **Sensitivity tests** 101
Sensitivity test conditions 101
Statistical analysis of sensitivity tests 102
Presentation of sensitivity tests 103
Interpretation of tests with regard to wind speed & direction 106
Interpretation of tests with regard to flight threshold temperature 108
Interpretation of tests with regard to flight time limiting 109

Chapter 6 Comments and conclusions
6.1 **General comments** 110
6.2 **Negative aspects of the simulation analysis** 111
6.3 **Positive aspects of the simulation analysis** 113
6.4 **Known flaws in the simulation model** 113
6.5 **Overall conclusions** 115
6.6 **Recommendations** 116

Alphabetical list of references cited 118

Appendix A Simulation test results: 1986-1989, Douglas, De Aar, Graaff-Reinet, Pofadder 123

Appendix B Rainfall plots: 1986-1989, Douglas, De Aar, Graaff-Reinet, Pofadder 136

Appendix C Statistical correlation tests applied to locust populations, simulated and controlled (killed). 138

Appendix D Interpolated data for holes in weather data set 140

Appendix E Simulation software Technical Reference & User Manual

Section E1 Getting started 141
- Typographical conventions
- System requirements
- MS-Windows Operating System shell
- Overview

Section E2 Running a trial simulation 146
- Loading weather data for a simulation run
- Commencing a simulation run
- Monitoring a simulation's progress
- Adjusting the duration of a simulation
- Retrieving the result of a simulation
- Executing a second simulation run

Section E3 Customising the simulation output 157
- How population records are stored
- How internal population variables become the output
- How to edit the monitor scripts
- How to use the output lists - simple output displays
- Understanding the "show" script
- Understanding the "nutab" script
- Understanding the "dumpwk1" script
- Understanding the "rain" script

Section E4 Customising the simulation input 170
- The model input
- The population variables
- List of population variables
- Customising the initial conditions
- When are initial conditions used ?

Section E5 Customising the model parameters 175
- Simulation model structural overview
- Changing the numerical values of lifecycle parameters
- Changing values used in weather data processing
- The degree-day contribution calculation
- The wet-index calculation
- Editing the rainfall classification boundaries
- The wind-index calculation
- Migration distance probability
- Flight time estimation
- Wind direction probability
- Combining probabilities
- Emigrating population estimates

Section E6 Adding your own / updated weather data 188
Basic procedure
Weather data required
Two types of data file possible
ASCII data file example
ASCII data file field specification
LOTUS 123 data file example
LOTUS 123 data file field specification
Adding a new previously unrecognised station
Step-by-step example

Appendix F Simulation model decision rules - quick reference 200

Appendix G Simulation population variables - quick reference 207

Appendix H Simulation parameters in effect during testing..... 212

List of figures

Fig 1.1	Interaction of institutions involved in the development of a locust EWS	3
Fig 2.1	The Brown Locust, *Locustana pardalina*	7
Fig 2.2	Schematic representation of population trajectories	10
Fig 2.3	Swarm ground velocities compared to wind speeds	15
Fig 3.1	Nested hierarchical model structure	35
Fig 3.2	Black box diagrammatic representation	37
Fig 3.3	The complete locust lifecycle block diagram	39
Fig 3.4	First-in, first-out (FIFO) queue schematic	41
Fig 3.5	Time of flight estimation - schematic	49
Fig 3.6	Source and destination sites: the loss of individuals to unmodelled sites	50
Fig 3.7	Interstation distance and associated Gaussian PDF	51
Fig 3.8	Probability that station B will be downwind of station A for any wind heading	55
Fig 3.9	Probability determined by the difference between omega and theta	56
Fig 3.10	AngularPDF: probability that staion A is upwind of station B	57
Fig 3.11	von Mises probability distribution	58
Fig 3.12	Graphical representation of migration engine application	60
Fig 3.13	Data interchange paths for increasing numbers of stations	62
Fig 4.1	Gantt chart of available weather and locust count data for potential sites	68
Fig 4.2	Examples of soil and air temperature differenced recorded at Douglas	70
Fig 4.3	Examples of air and soil temperature maxima and minima recorded at Douglas	72
Fig 4.4	All outbreaks recorded for the period 1985..1992 at sites of interest	77
Fig 4.5	South Africa: locations of sites of interest	78
Fig 4.6	Maximum difference between two hypothetical CDFs	91
Fig 4.7	Scatter plot of two random variables	92
Fig 5.1	Venn diagram: variable definitions for Jaccard, Simpson, Dice and correlation coefficient	100
Fig 5.2	Sensitivity to β in regard to flight distance probability	105
Fig 5.3	Sensitivity to b in regard to flight direction probability	106
Fig 5.4	Sensitivity to flight threshold temperature	107
Fig 5.5	Sensitivity to flight time limiting	109

Figures contained in the appendices:

Fig E1.1	Windows program manager, showing possible icons including the simulation model icon	143
Fig E1.2	Simulation Model Banner Window	143
Fig E1.3	The windows task manager, used to select foreground applications	144
Fig E2.1	Control panel dialogue box for the simulation modelling software	146
Fig E2.2	The data loading dialogue box	147
Fig E2.3	Simulation block diagram	152
Fig E5.1	Hierarchical structure of the simulation model	176
Fig E5.2	Simulink / model block diagram parameter editing dialogue box	178
Fig E5.3	Definition of degree-day calculation	180
Fig E5.4	The shape of the p(s) curve	184
Fig E5.5	Flight time estimation curve	184

List of tables

Table 4.1 Data distribution of locust outbreaks 84
Table 5.1 Scatter plot point counts of simulation results against actual outbreak events 99

List of symbols

The [illegible]owing acronyms and abbreviations appear in this document:

ARC ..... Agricultural Research Council
CPU ..... Central Processor Unit = microprocessor
CCWR ..... Computing Centre for Water Research (Natal)
DALR ..... Dry Adiabatic Lapse Rate
DRC ..... Directorate of Research Conservation
ELR ..... Environmental Lapse Rate
EWS ..... Early Warning System
FAL ..... Free Air Layer
GIS ..... Geographical Information System
ISCW ..... Institute for Soil, Climate and Water (Pretoria)
KB ..... Kilobyte = 1024 bytes
LLJ ..... Low Level Jet
MB ..... Megabyte = 1024^2 bytes
mb ..... milliBar
MHz ..... MegaHertz = 10^6 cycles per second
NOAA ..... National Oceanographic & Atmospheric Administration
NDVI ..... Normalised Difference Vegetation Indicies
NWS ..... National Weather Service (of the USA)
OOPS ..... Object Orientated Programming Structure
OS ..... Operating System
Pa ..... Pascal
PC ..... Personal Computer
PBL ..... Planetary Boundary Layer
PPRI ..... Plant Protection Research Institute (Pretoria)
RAM ..... Random Access Memory
RMS ..... Root Mean Square
SAWB ..... South African Weather Bureau (Pretoria)
SBL ..... Surface Boundary Layer
WALR ..... Wet Adiabatic Lapse Rate

Chapter 1 OBJECTIVES

1.1 TOWARDS DEVELOPING A LOCUST OUTBREAK EARLY WARNING SYSTEM

During 1994, the Agricultural Research Council (ARC) convened a multidisciplinary task group, to work together on a project titled "Towards developing an early warning system to predict Brown Locust outbreaks". The following organisations have been intimately involved in the project:

The Agricultural Research Council, with representation from the following ARC departments: The Plant Protection Research Institute (PPRI),
The Institute for Soil, Climate and Water (ISCW),
The GIS (Geographical Information System) Department,
Remote sensing,
Agromet.

The University of the Witwatersrand, specifically the Department of Zoology.

The Department of Agriculture, and the Directorate of Resource Conservation also play a rôle.

The specific aims of the enterprise are outlined as:

- Collection and collation of locust data in the outbreak region
- Retrospective analysis and mapping of locust outbreaks
- Improved locust monitoring systems
- Analysis of weather data in the outbreak region
- Bio-modelling of locust outbreaks
- Developing an early warning system (EWS) using weather data, vegetation indices and locust data, and integrating this into a Geographical Information System.
- Providing rapid exchange of information on past outbreak status to achieve more efficient management and deployment of resources.

The University of the Witwatersrand's Department of Zoology has been concerned with the weather data analysis and bio-modelling of locust outbreaks.

The rôle of the Zoology Department has been one of providing some of the biological foundations - in collaboration with members of the PPRI - for building the locust lifecycle model. In addition to being a source of information of the lifecycle parameters, the Zoology Department has been responsible for collating this information with that from the PPRI, and for the software development and coding of a fully workable computer simulation, the details of which are described in this document.

The operation of these simulations hinges on environmental data obtained from the PPRI, the ISCW, and the S.A. Weather Bureau. Additionally, the ARC requested that the simulation model require only data fields which they could supply from their or the SAWB's databases.

POTENTIAL IMPACT OF THIS RESEARCH

The Brown Locust is a serious insect problem in South Africa. An indication of the magnitude of this problem is reflected in the fact that R11m[1] was spent on locust control during the 1993-1994 season. This pales in comparison to that spent during the previous major outbreak. Toms (1988) reported an expenditure of R28,3m for the 1985-1986 Karoo outbreak season, and declared that the locusts were not considered to have been successfully controlled. These figures do not indicate indirect costs such as environmental pollution. The real financial impact is reflected in Toms' analysis of return per hectare.

He points out that R28,3m translates into an approximate cost of R5,66 per hectare, yet the average net profit recorded by typical Karoo farms over the three years preceding the outbreak amounts to only R4,51 per hectare. Further, during the period 1907 to 1967, there were only six years during which there were no *reports* of brown locusts.

An accurate EWS - based on sound scientific data - would alert decision makers to the danger of impending outbreaks. A better management support system would exist for timeous planning and execution of control campaigns. Limited resources could also be better utilised in response to rapidly changing priorities. Targeting specific areas would reduce the cost involved in poison distribution and recovery. Additionally, better estimates of the required amounts would be possible. In essence, the problem to be solved is delivering the right amount of poison to the right place at the right time.

The proposed project will utilise and integrate the latest technology, such as GIS, satellite-based vegetation indices, the AgroMet databank, and the ARC's wide area network (WAN) and other computer technology.

The ARC, and more specifically the Agromet section of the ISCW, has been the primary co-ordinating body of the project. Agromet is also responsible for supplying weather data to the group, as well as analysing weather data in the Karoo. Regular meetings and workshop sessions have enabled all the *contributors to discuss the availability of essential information*, and the shortcomings where those arise.

A diagrammatic representation of how the different institutions interact is shown overleaf in Figure 1.1.

1 Agendum - meeting of involved parties - March 1995, ARC headoffice, Pretoria.

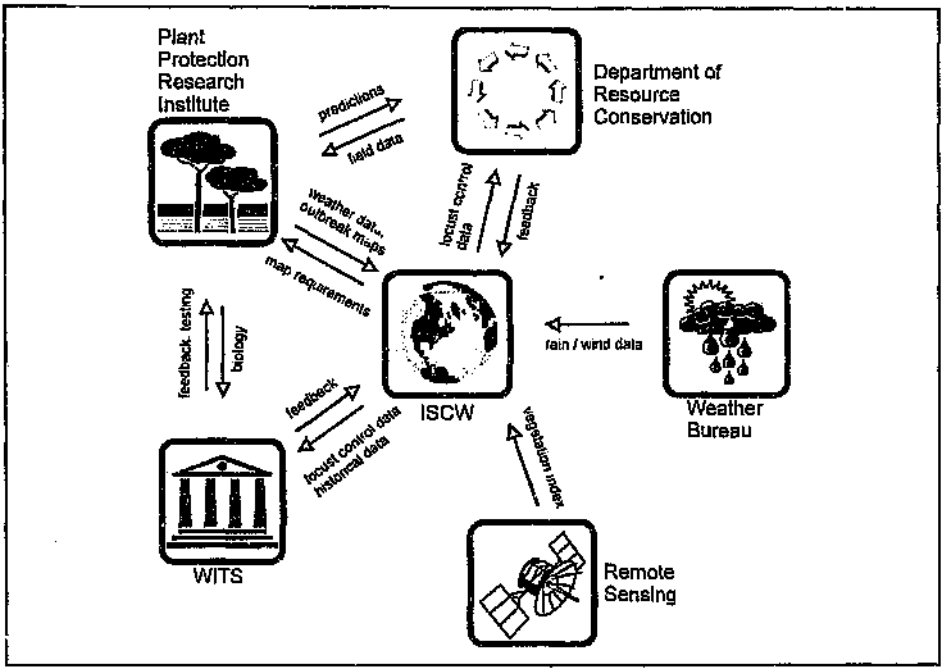


Fig.1.1 The interaction of institutions involved in the development of a locust early warning system. The contribution made by WITS is described fully in this document.

DYNAMIC SIMULATION MODELLING

The simulation model developed comprises two more-or-less independent components. The core of the simulation is a software model of the Brown Locust lifecycle. This lifecycle simulation has been used as a building block in the development of a migration simulation model that tracks the movements of locust populations in response to changing weather conditions.

The logical conclusion is to integrate the locust outbreak simulation model into the EWS, which is based on a UNIX™ operating system. The simulation model at present runs exclusively on a personal computer (PC™), using the Mathworks™ Matlab™ command interpreter, which runs under Microsoft™ DOS™ and Windows 3.1™. As such it does not interface directly with the proposed GIS-based early warning system. However, the existing model provides two essential cornerstones of the EWS:

Firstly, the PC-based simulation model provides a platform for testing the theory and lifecycle parameters. A greater degree of testing and experimentation can be done since operators are not confined to limited-access, high-cost GIS-compatible hardware.

Secondly, Matlab is available for the UNIX™ operating system. At some future date it may be desirable to transport the model to this operating system. Since the Matlab interpreter syntax is universal to all operating systems (except for the graphical capabilities of that system), the translation is a simple, fast and cheap process.

The project objectives which are specifically relevant to the current author, and which are addressed in this document are:

- Is it possible to determine migration movements from wind speed and wind direction? Can this be done accurately enough in order to achieve the objective outlined directly below?
- Will the accuracy of outbreak predictions improve significantly if the simulation model is expanded to take migration movements - based on wind phenomena - into account?
- Will taking migratory movements into consideration create suitable initial conditions for the next season? This differs subtly from the objective directly above in that it will take into consideration migratory effects, such as population relocation, and dispersal of eggs over many seasons. The question addressed above is directed more towards finding out if population redistribution has an immediate effect.

The experimental hypothesis may be summarised as:

Incorporating the phenomenon of wind-facilitated migration into the basic one-point lifecycle model will increase the accuracy of the predictions on a temporal scale (i.e., simulated outbreaks will coincide more precisely time-wise with actual outbreaks).

1.2 THE SCOPE OF THIS DISSERTATION AND WORK ACCOMPLISHED

The initial scope of the enterprise was limited to extending an existing simulation model (Nailand and Hanrahan, 1993; Hanrahan and Hanrahan, 1994). The model was to be modified to incorporate the necessary software entities so that it could simulate locust migration between various weather stations.

The simulation model, as it stood towards mid-1994, proved to be completely unworkable. Encumbered by unnecessarily complex and slow software code, and excessively greedy with regard to computer memory, it failed to provide a useful platform on which to build the migration model.

The entire software suite comprising the model was re-written, paying careful attention to the following factors:

- The revised model should be at least as easy to use as the original version, as far as the user interface is concerned. This would facilitate a rapid changeover for people already familiar with the operation of the older model.

- The revised software must be as frugal as possible with regard to computer resources such as memory (RAM).
- The revised software must execute as fast as possible.

With regard to the second two points, speed and compactness were of the utmost importance. No matter how fast, or memory-frugal the software is, it is always desirable to improve the specification of both of these parameters.

The lifecycle model (Nailand & Hanrahan, 1993), after improvement, was used as a basic building block in creating the larger scale migration model developed by the current author. This migration model simulates locust population fluctuations at two (or more, if desired) geographical locations, and simulates movements of migratory swarms between these sites.

An additional objective of the reworked software was, while retaining bac...ard compatibility with earlier commands, to make it very simple to operate and update, so that potential users need have no special modelling knowledge or programming skills. To this end, the model now operates through a standard Windows™ graphical interface, with menus and dialogue boxes. All data input and output is in industry standard spreadsheet format, and several graphical utilities and statistical tools have been added to make interpretation of the results quick and easy.

A comprehensive technical user manual has been written in order to facilitate use of the model by those with little experience of computer programming or simulation modelling. It does, however, include an entire technical reference explaining all the internal operations of the simulation model. Such information will prove invaluable to those wishing to modify or expand the software at some future date. This document is included as Appendix E. Nevertheless, the simulation model remains a moderately complex entity, and is not intended to be a use-it-out-of-the-box product for casual users who do not have some understanding of computer basics or modelling theory.

Various other software utilities were developed to complement the simulation model. These included programs to convert various formats of weather data, and to analyse the simulated results, but since their development bears no direct relation to the simulations, they will not be discussed further.

The following chapter deals with the agricultural and biological background to the problem of locust swarming. A simple population model is discussed, and the concepts required to expand that model to a multi-point migration model are introduced. The rôle that wind plays in driving migrations is addressed with a view to developing the mathematical representations required to implement to simulation software.

Chapter 2 BIOLOGICAL and AGRICULTURAL BACKGROUND

2.1 THE NEED FOR A LOCUST OUTBREAK SIMULATION MODEL

The Brown Locust - *Locustana pardalina* (Walker) - regarded by Matthèe (1951) to be "undoubtedly the most important of the African locust species" is also considered by Uvarov (1928, 1966) to be the chief destructive species of locust in South Africa. This opinion is strongly supported by the fact that the insects have been declared a national pest - making control of swarms a legal obligation to the relevant government authorities (as reviewed by Hanrahan, 1988). The Brown Locust regularly forms swarms with a periodicity of seven to eleven years, although the actual period within this interval is highly variable, making prediction difficult. During the last two-hundred years or so that interest has been taken in the Brown Locusts' population dynamics, a change has been noted in the general periodic trends. Although formal recordings of outbreaks have been made for South Africa as a whole only since 1910, De Villiers (1988) has noted that up until the second decade of this century, the duration of plague events was approximately thirteen years, interspersed with quiet, non-plague intervals varying about a mean of eleven years. Prior to 1907, population control of locusts was characterised by natural physical means, namely predation. Lea (1968) observed that since various chemical means have been adopted with regard to controlling locust numbers and plague outbreaks, control bodies have neither succeeded in delaying the regular onset of such plagues, nor have they shortened their duration. The only tangible evidence that chemical control has had any effect, is an apparent shortening of the inter-plague period to around seven to eleven years, and Lea (1958) notes that control measures introduced in the late 1940s have been able to confine locusts to their outbreak region.

The Brown Locust constitutes a major threat to agriculture and suppressive control in the outbreak area requires considerable expenditure (Botha 1967). Toms (1988) reports an expenditure of mR28,3 for the period 1985 to 1986 for locust control in the Karoo, and that the locust swarms were not considered to have been successfully controlled. This period reflects the most recently recorded instance of major swarm activity (not including 1994 through 1996) for which comprehensive information is available, although Toms also reveals that during the interval 1907 to 1967 there were only six years in which there were no *reports* of Brown Locusts. He points out the general assumption that if incipient swarms were to get out of control, the expenditure could be much greater, but provides no numerical figures to back up this statement. Further, he notes that there are no accurate figures to indicate indirect costs such as environmental pollution. Wiese and Bot (1973) have suggested that the high level of γ-BHC (gamma benzene hexachloride) that was found in humans in research done in the late 1960s was derived from that used to control locust outbreaks in the Karoo. The use of γ-BHC has since been banned in South Africa due to its ecologically harmful effects,

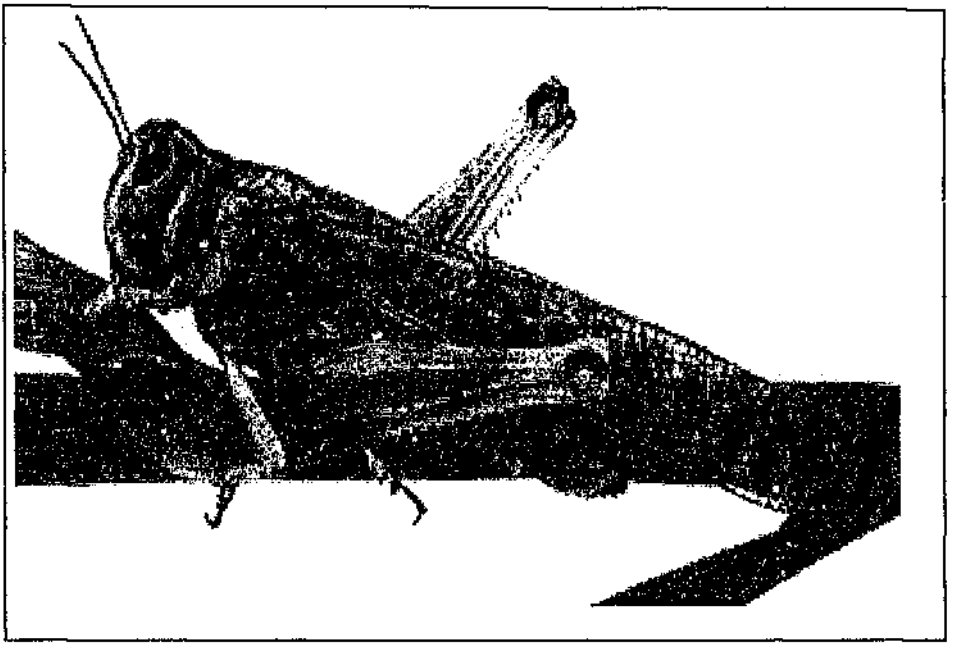


Fig. 2.1 The Brown Locust, *Locustana pardalina* (Walker)

although it is still considered an acceptable poison solution in many other African countries. Remarkably enough, Hill (1983) actually recommended γ-BHC as the poison of choice for locust and grasshopper control.

Toms (1988) theorises that if locust research is proportional to the immediate threat, then the quantity of research will appear much like a classical predator / prey time analysis graph.

This suggests that control bodies are best equipped to solve the problem when it is in its natural decline phase or when it is no problem at all. Conversely they are least equipped to deal with the problem in its exponential growth phase. He suggests that an effective control strategy might be most appropriately applied in the middle of a trough - or cessation - when the problem is hardly apparent.

This concurs with Hanrahan's 1988 comments that during the preceding twenty years, locusts have represented a comparatively minor problem in Africa, leading to the conclusion that any lessons learned during earlier plagues have gone largely unheeded.

Wagner & Willers (1992) summarise two common land management approaches to dealing with insects: through direct control at times of crises (during outbreaks), or through direct control at times when no control is needed (endemic populations). These strategies they refer to as crisis-orientated management and fear-orientated management respectively, both of which provide only a limited success rate. The problems arise from a scarcity of knowledge, or the inability to access and effectively use existing knowledge.

Personal communications the Department of Agriculture (Heyns, 1996, pers. comm.) have revealed that their main problem, when planning anti-locust campaigns, is estimating the magnitude and duration of a swarm event This often results in too little poison being made available - the consequence of which is a proliferation of insects in the absence of adequate control measures. The converse situation - overproduction of poison - means costly programmes to recover unused poison from farmers and local storage depots, and the need to store or dispose of these dangerous products. Careful control of the poisons is essential since many farmers fail to realise the extreme toxicity of the product and do not adhere to adequate safety measures.

An accurate method of predicting locust outbreak activity would allow more efficient control resource usage. Targeting specific areas would reduce the amount of poison released into the environment, as well as the cost involved in its distribution, and recovery.

Predicting major outbreaks of locust activity is thus a highly desirable talent, and many attempts to achieve more accurate forecasting techniques have been made, both locally (du Plessis, 1938; Smit, 1941; Nailand & Hanrahan, 1993; Napier-Backs, 1996, pers. comm.) and abroad (Farrow & Longstaff, 1986; Dennis, Kemp & Beckwith, 1986; Krebbs, 1978; Wright, 1987; Berryman, 1987; Risch, 1987; Day & Knight, 1995; Robinson, 1995; Lecoq, 1995).

Smit (1941) arrived at some general maxims for predicting a high probability of an outbreak. These were based on meteorological conditions such as a minimum rainfall during the summer months, but only after one or more dry seasons had occurred. Smit proposed somewhat different criteria for rainfall to Du Plessis (1938) yet is substantially in agreement with Du Plessis as regards those environmental conditions conducive to imminent swarming activity. Price (1988) expounds on these basic theories, suggesting that at the onset of spring, with sufficient rainfall, almost all eggs in the soil will hatch simultaneously. This factor, in conjunction with his proposal that outbreaks occur after a build-up of solitarious females during the previous seasons, can account for the observed trends. Mattheè's (1951) findings that solitary females lay proportionately more diapause eggs than gregarious females, and that proportionally more diapause eggs are laid as the female ages, certainly lend support to this hypothesis.

A BASIC POPULATION MODEL

Nailand and Hanrahan's 1993 work on a Brown Locust population model provided the basic building blocks from which the locust lifecycle model has grown.

May (1976) provides insight into the mathematics associated with various modes of population growth and has determined certain prerequisite conditions necessary for chaotic population fluctuations. In essence, he shows that a Leslie matrix may be used to compute the intrinsic rate of growth within a population, and determine whether chaotic behaviour is a justifiable explanation for the observed trends. Cheke (1978) provides comprehensive guidelines for the use and solution of such Leslie matrices. Cheke's methods were also adopted by Farrow and Longstaff (1986) in determining the necessity for an explanation other than the

intrinsic rate of increase to precipitate plague events. Recently, Higgins, Hastings, Sarvela and Botsford (1997) have shown that erratic population fluctuations in the Dungeness crab are generated by density-dependent mechanisms that are both non-chaotic, and produce a strongly stable equilibrium in the absence of exogenous stochaticity.

While the original locust lifecycle model of Nailand and Hanrahan (1993) appeared sound from a biological point of view, factors other than the intrinsic population growth rate are clearly at work, and need to be incorporated into the lifecycle model if the key to plague initiation is to be understood. The first logical step in continuing the lifecycle model, was to consider the rôle of migration, a prominent characteristic of locust swarms and upsurges (Pedgley, 1979).

In brief, the migration model simulates the waxing and waning of (two) different populations of locusts - at various stages of development in the lifecycle. These populations are presumed to be distinct, apart from migration of individuals between them and other such distinct populations. Interactions between these population entities are then considered and simulated using a wind-based trajectory model, developed specifically to interface with the lifecycle model. The core building block of the migratory simulation model is the locust lifecycle model.

A graphical representation of such a trajectory model is shown in Figure 2.2 overleaf. Note that in this simplified representation, fluctuations in local population size are not shown.

2.2 ATMOSPHERIC PHENOMENA AND INSECT TRAJECTORIES

Insect migration modelling requires the integration and standardisation of large, complex and varied sets of cross-disciplinary information. Data containing such diverse information as (nearly) continuous weather records, discrete counts of trapped or killed insects - which are sometimes only estimates - requires merging into meaningful and compatible data sets.

TAKING POPULATION MIGRATIONS INTO ACCOUNT

It is well known that atmospheric phenomena are an influence on locust migrations, but certain specific features of the atmospheric environment have been found significant in relation to the distribution and movement of pest populations, and in relation to forecasting the incidence of migrant pests (Drake & Farrow, 1988; Farrow, 1990). Further, the long distance transport of such migrants by large-scale wind systems frequently initiates outbreaks of agricultural pests in the destination region. Insect migration can be affected by a wide variety of meteorological phenomena and towards the end of the last decade new research revealed how migrants interact with *small-scale* weather features and in particular, with the vertical structure of the atmosphere (Drake & Farrow, 1988). It has also been noted that insect migratory flight is confined to the lowest two kilometres of the atmosphere, and in the case of the Brown Locust, a substantially smaller band even than two kilometres (Price, 1996, pers. comm.). This two

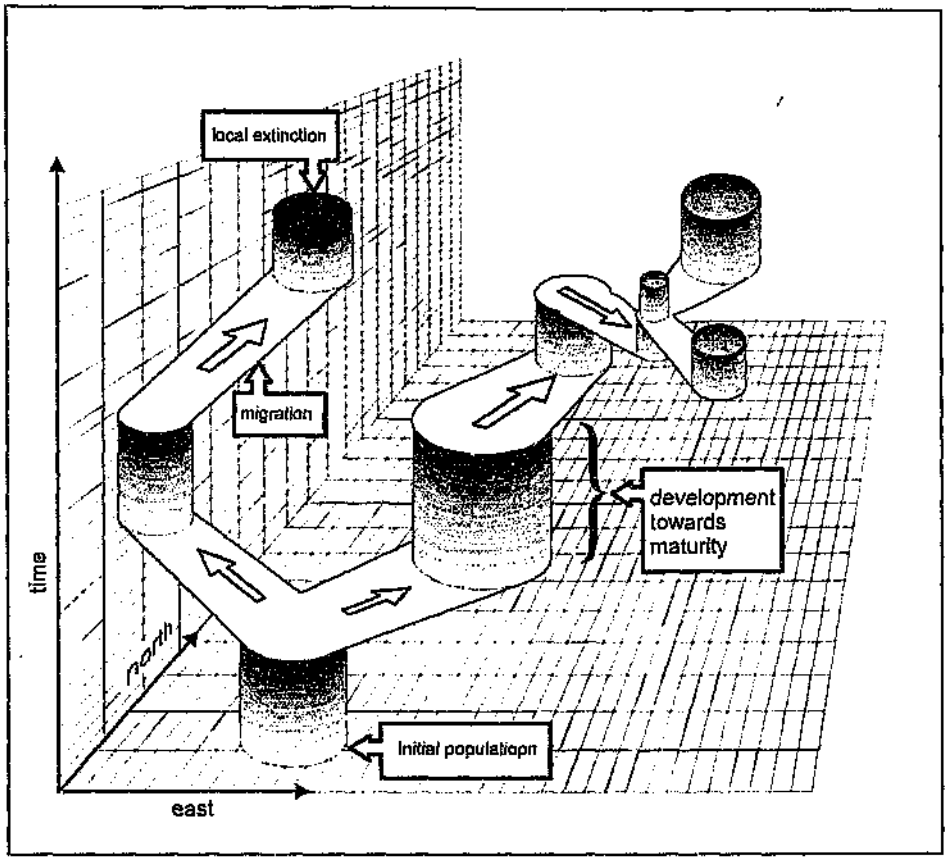


Fig. 2.2 Schematic representation of general hypothetical population trajectories, showing migration and subsequent growth or extinction. (After Drake & Farrow, 1988)

kilometre region coincides with the *planetary boundary layer*, defined to be that zone in which the underlying surface has a direct effect on the atmosphere. The planetary boundary layer can exist in one of two states: convective, and stable. Migrating insects are strongly influenced by the planetary boundary layer in both forms (Drake & Farrow, 1988).

Farrow (1990) relates that flights by insects in the upper air were previously considered to be largely accidental, resulting in passive transport by wind to destinations over which the migrants had no control, and where most failed to reproduce. Farrow proclaims that it is now widely recognised that migration is an active process by which [Acridoids] achieve significant spatial redistribution. This proposition is supported by Rainey (1989).

The wind speed in the lowest few tens of metres (defined to be the *surface layer*, a zone of particular interest in the case of Brown Locusts) increases as a near logarithmic function of altitude. Beyond the surface layer the speed continues to increase, but the wind direction becomes influenced by factors independent of the

surface features, specifically the Coriolis force. Towards the top of the planetary boundary layer the wind is approximately geostrophic[1]. This phenomenon gives rise to what is known as the Eckman spiral, a description of the wind bearing as a function of altitude. The failure of the wind to blow directly down the pressure gradient is a consequence of the planet's axial rotation.

Farrow (1990) has suggested that nocturnal temperature inversions are particularly favourable for sustained long period flight, even long after the surface air has cooled below the threshold for flight. These conditions correspond to the stable form of planetary boundary layer (increase in temperature with altitude).

The wind profile of the stable planetary boundary layer frequently exhibits a wind speed maximum at an altitude of several hundred metres. This effect is known as a low-level-jet. Drake and Farrow (1988) maintain that low level jets that occur in the absence of cold fronts or topographical features such as coastlines and mountain ranges, are found only at night over inland areas that are relatively flat (such as the Karoo). The top of the nocturnal wind shear occurs just below the top of the temperature inversion boundary layer where wind maxima and low-level jets are often found (Farrow 1990). Brown Locusts are not usually observed in high altitude night-time migrations. However, if the model is adapted for use in other situations, this factor may prove important. Seino *et al* (1987) in Wantanabe (1995) obtained highly significant correlations between the immigration of planthoppers into Japan and the development of low-level-jet airflows (LLJs). Wantanabe has developed a computer program to identify LLJs, with the objective of monitoring planthopper immigration. The program is based on wind speeds and directions measured at the 850mB pressure surface.

Wind trajectories derived from real time patterns of atmospheric circulation can be used to represent regional displacement of acridoid insects (Pedgley 1981 in Farrow 1990). At night it is assumed that they are transported downwind in the nocturnal geostrophic wind, details of which are confirmed by Drake & Farrow (1988), at a speed exceeding the streamline by up to 25kmh^{-1}. In some instances, supergeostrophic flows develop that can transport the insects even faster than expected. Such features are not always detected by standard meteorological observations. (Farrow 1990). Emphasis is placed on real-time streamline charts, in contrast to those which show the general seasonal trends.

Farrow states that in southern Africa, the northward spread of plagues in the first generation of the new season of Brown Locusts does not accord with the expectation of the downwind movement to the south on pre-frontal airflows, observed [in July]. He continues to say that movement could occur on a southerly anticyclonic [high pressure cell] flow, although that flow would be relatively cool at that time of year.

1 geostrophic wind: air flow with a direction parallel to isobars and perpendicular to the pressure gradient: occurs in upper atmosphere where flow is relatively independent of surface influences.

This factor would have a negative influence on the insects' flight due to their flight threshold temperature - an important factor. According to Smit (1939) easterly movements predominate in the summer generation, which are decidedly against observed airflows. Farrow's explanation of this phenomenon is that the swarms are generally low flying, and the preferred direction is maintained independent of wind direction both in spring and in summer. This contradicts Wood's 1956 measurements in east Africa (Rainey, 1989). These movements would take swarms into areas of summer rainfall where chances of survival are likely to be higher. However, analysis of the records for the 1986 outbreak in South Africa reveals that the greatest densities of hoppers are found in the months centred on March and September, not in the periods centred on July and January. These midwinter and midsummer months appear to be the periods upon which Farrow (1990) has based his assumptions, and contradict his earlier proposal that cause-effect relationships should be based on real-time streamline flows and not general trends.

Furthermore, Farrow's illustrations incorporate generalised synoptic weather patterns, which by convention show wind patterns at the 850mB pressure surface. This typically exists at a height of approximately 1500m, which as he points out, is much higher than the typical flight altitude of the locusts. He does, however, conclude that most nocturnal migrants are displaced downwind in the geostrophic flow, as mentioned earlier.

Records of near-ground wind speed and direction would provide a more suitable reference from which to test any hypothesis relating wind to migratory movement. As noted previously, Drake and Farrow (1988) surmise that the zone of insect migratory flight extends in instances to 2000m, which indicates that note should be taken of both the synoptic wind patterns as well as ground level measurements.

INCORPORATING WIND EFFECTS INTO THE MODEL

Drake and Farrow (1988) provide - for each of the atmospheric features shown to affect insects during migratory flight - a brief description of the phenomenon and a description of the entomological observations. Since it is desirable to transport the simulation model to a grid-base, it will be necessary to determine how individual cells of the grid interact. Reliable recording of swarm movements has not been done and wind phenomena are suspected of providing a reasonable indicator of migratory movements. This suggests that incorporation of wind speed and wind direction into the model should be done, where a certain fraction of individuals would be removed from the population if they were of sufficient maturity, and if the wind speed was sufficiently high.

However, there is also the limiting factor of low air temperature - a function of increasing altitude - which constrains the insects' ascent beyond a certain altitude - that must be taken into account. Rainey (1989) describes the significance of this for the development of thermal convection currents, and in turn the vertical distribution of airborne insects. Numerically, the temperature decrease as a function of altitude is described by the dry adiabatic lapse rate (DALR) which is

found empirically to be approximately 9,8°C/km. Also features of the atmospheric temperature profile are the wet adiabatic lapse rate (WALR) and the environmental lapse rate (ELR), although Rainey does not make mention of these. The magnitude of the DALR is a constant. The figure of 9,8°C/km is very relevant if one considers the claims that insect migrations occur at altitudes up to 2000m (Drake & Farrow 1988), since it implies temperatures at that altitude can be near to, or indeed below 0°C. If, on the other hand, Price (1996, pers. comm.) is correct in his assumptions that Brown Locust migrations occur at altitudes of around 10 to 20 metres, then the lapse rate implies a drop in temperature of only 0,2°C over that height, which is no cause for concern, since internal mixing within the surface layer will mask the lapse rate effect.

Baker *et al* (1984) have observed that the swarms of Australian migratory locusts most often occur in the 6,0m to 6,2m above ground altitude band. However, this does not imply that this is the predominant zone for migratory flight. As noted by others, there is evidence to suggest that the migrating locusts occur in a belt, or belts at much higher altitudes - the geostrophic wind zone. At such altitudes (1km to 2km) there is little opportunity for viewing them directly from the ground.

Farrow (1990) indicates that it is difficult to show that flight has been terminated by falling air temperature. Drake and Farrow (1988) intimate that although insects may actively avoid ascent, they may be carried up to altitudes where the temperature is below their flight threshold and then their failure to ascend may be simply passive. This is important, they say, since such insects failing to reach the zone of divergence above a rising thermal circulation will collect at the region of convergence[2] at the base [of the circulation] and not be transported laterally.

Drake and Farrow (1988) noted that locust flights began at dusk with mass take-off, and continue throughout the evening until approximately midnight. They suggest that take-off is initiated by falling light intensity, and only occurs if the temperature is greater than approximately 20°C. Sustained migratory flight, but not necessarily take-off, is inhibited if the immature locusts have been unable to accumulate adequate energy reserves, owing to lack of green feed. (Symmons and McCulloch, 1980, in Drake & Farrow, 1988). Botha and Jansen (1969) put forward that "the old world species of locust" [most likely the Desert Locust, *Schistocerca gregaria*] characteristically fly during the day during the gregarious phase, and roost at night. Conversely, in the solitary phase, they are usually sedentary by day but may make considerable flights at night. This Botha and Jansen support by quoting Waloff's 1963 findings that flight [of the Desert Locust] may be a regular nightly occurrence in low density solitarious populations.

Adult Brown Locusts are sometimes found in large numbers at street lights in Karoo towns in the outbreak region, even when there is no observed swarm activity. Botha & Jansen (1969) claim that this indicates that night flights of solitarious insects do occur on a substantial scale.

---

2 Thunderstorms result from the convergence of air in depressions or troughs of low air pressure (cyclone systems). Such convergence leads to rising air currents. Where sufficient moist air is involved and where conditions are unstable, convergence provides the trigger action for thunderstorms. *Divergence* involves the descent and adiabatic heating of air in high pressure cells (anticyclones), and is associated with fine weather conditions.

However, an alternative explanation lies in the locusts being actively drawn by the presence of the lights. In other words, the lights induce flight activity that would otherwise be absent (Hanrahan, 1998, pers. comm.).

MIGRATION SOURCES AND SINKS

In a process such as migration, two pieces of information are of paramount importance. The source of the migrating swarm must be identified, and its probable destination site must be identified - or estimated as accurately as possible. These, in essence, define the migration. Time, as will be shown later, is closely linked to distance and is thus accounted for by specifying location.

Kennedy (1951) in Baker *et al* (1984) suggests that to the locust, the apparent movement of a fixed background is an unsettling stimulus which provokes a turning response. This may be either into the wind - if the wind is light, or down-wind if the wind is strong. The objective of this being for the locust to maintain a certain preferred speed of ground moving under it. Kennedy also remarks that there is a point at which the apparent ground speed is so low that the locust can no longer respond to it. This will occur when the locust is above a certain height - the so-called *maximum compensatory height.* This implies that locusts moving into the geostrophic flow have little or no ability to gauge their own speed, and will allow themselves to be carried by the wind regardless of its direction or speed. There is also the unadressed matter of whether surface features offer suitable visual cues during night flights.

Uvarov (1977) claims that individuals and solitary locusts do not fly spontaneously by day from low density populations, but Farrow (1990) contradicts this, proposing that daytime flights of Acridoids at (relatively) high altitudes of 110 - 300m and at speeds of greater than 20kmh^{-1} have been recorded. Such flights often result in migratory movements between habitats. This partly concurs with observations made by Price (1996, pers. comm.), although Price suggests lower altitudes for the Brown Locust. Farrow claims that more important are the night-time flight movements, since these are associated with large scale migratory movements, when nocturnal migration is characterised by remarkably uniform and persistent cross-country flights, typically at several hundred metres of altitude. Flight is initially upwind, but if the wind speed is $\geq$2ms^{-1}, protracted flights are predominantly downwind, typically at heights of 150 - 500m. (Farrow 1990). The zone in which the insects' flight speed is lower than the wind speed is referred to by Drake & Farrow (1988) as the flight boundary layer (FBL). They also confirm that migrations above the surface layer are generally close to downwind, because the wind speed is usually high compared to the flight speed.

Rainey (1989) provides a concise graphical representation of the observation made in east Africa during the 1950s. This graph is reproduced overleaf in Figure 2.3. It shows the relationship between locust orientation, overall insect velocity, and wind velocity. In essence, the net locust velocity is largely determined by the wind, although there appears to be a constant difference between these two velocities due to the non-coherent orientation of individuals within a

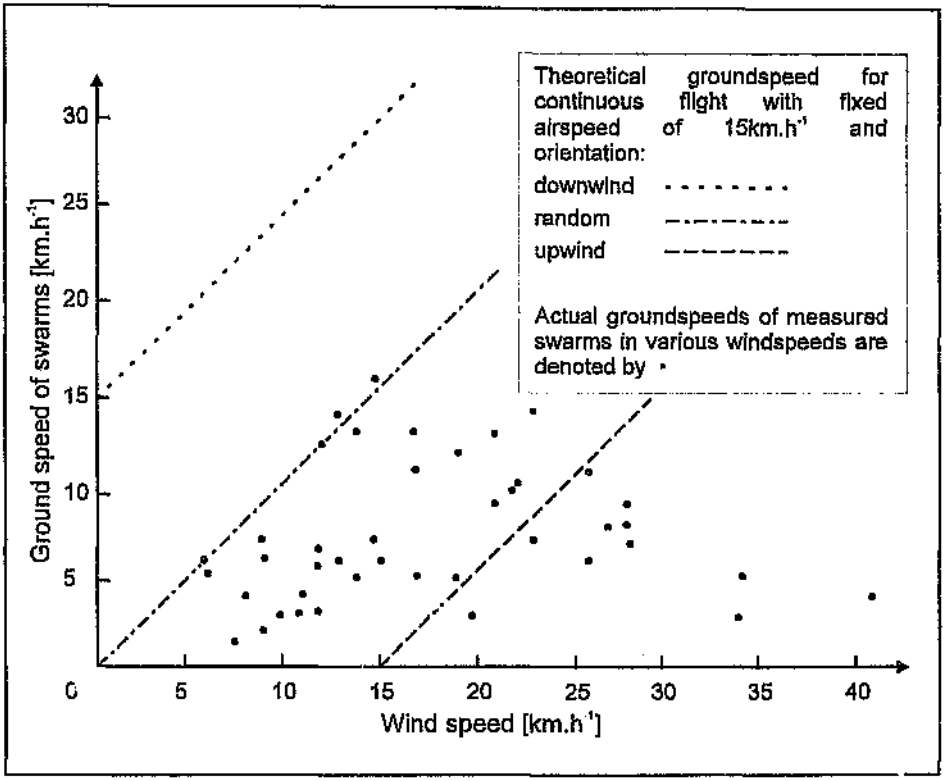


Fig. 2.3 Swarm ground velocity compared to wind speed. (After Rainey, 1989)

swarm. Commonly occurring insect velocities range from 20 - 50kmh^{-1}, and typical flight duration from 3 - 5 hours. Drake and Farrow (1988) suggest flight durations from <1h to approximately 10h. This can result in an overall displacement of up to 250km in a single night (Farrow 1990). Rainey (1989) claims flights of up to 40hours at speeds of 15kmh^{-1} for the Desert Locust, which corresponds to a total flight distance of 600km. This is not necessarily the overall displacement of the swarm though, since direction may change during flight and so may locust orientation.

The number of migrants in such movements depends primarily on the size of the contributory source, and not on the immediate weather conditions (Johnston (1969) in Farrow (1990)). Rainey (1989) mentions swarms that were estimated to contain 10^9 (a thousand million) individuals, and extending over an area of 30km^2.

Farrow (1990) also indicates that probable triggers for migratory flight are temperature, humidity, wind speed, rainfall and air pressure. Large scale takeoffs are rare if temperatures are below 20°C. In the subtropics, migratory activity is often enhanced by the occurrence of atmospheric instability (Farrow, 1990; Drake & Farrow, 1988; Rainey, 1973) which is associated with low level convergence. Rainey (1989) observes that wind speeds exceeding 40kmh^{-1} are not conducive to low level flight, and thus will inhibit take off, although at higher altitudes (up to 600m) no such flight inhibitions are observed. Note that high velocity winds at high

altitudes do not necessarily imply such high velocities at the surface. In fact, sudden changes in wind velocity and direction with altitude are characteristic of the wind shear (Farrow, 1990). Drake & Farrow (1988) indicate that migrating insects are likely to become concentrated in a zone of convergence that exists for any length of time.

Lambert (1972) reiterates the fact that night-time flights tend to be of a migratory nature. Night flight involves mainly older insects, but only those which are still sexually immature or females that have recently oviposited and contain poorly developed oocytes. Locusts with fully developed eggs do not fly at night, which is substantiated by Clarke (1971), indicating that migratory insects will not have a substantial contributory influence on the local eggbeds (migrating individuals will only reach sexual maturity after they have migrated; their contribution to the egg beds will be at another location). Clarke confirms that observations of night flight agree with the view that migration commonly occurs in sexually immature insects after a teneral period, during which flight patterns become fully developed. Clarke (1968) has also shown that cycles of oviposition [in the Australian plague locust] can take place every twelve days, depending on conditions. If locusts carry out prolonged flight activity only when they are sexually immature or when they have recently oviposited, then each individual would be expected to fly for only a few days in every twelve. Hence only a portion of the population could be expected to migrate on any one night. That proportion would depend on how many have oocytes in these classes [<0,89mm in length] at that time. In order to reasonably assess the likelihood of any population undergoing migration, the maturity of the adults would have to become a monitored parameter. Botha and Jansen (1969) concluded from their observations that the occurrence of night flights must be associated with the age of the adults and not only favourable weather conditions.

In determining the fraction to be removed from a particular population, it is necessary to take into account this postulation by Botha and Jansen. Clarke, Ashall, Waloff & Chinnick (1969 in Lambert, 1972) recorded an estimated average of 35% of locusts leaving observation areas per day during work done in Queensland.

PRACTICAL MODELLING CONSIDERATIONS

A synopsis of the nature of various wind phenomena has been provided. In many cases, the interaction between insect flight and the wind is more speculation than fact, making the exact rôle of the simulation engine somewhat nebulous. In designing the simulation model - as an expansion of the lifecycle model - an attempt has been made to incorporate generally accepted theory, but to avoid modelling processes which are more rare, or less well understood. Retaining model simplicity is an underlying prerequisite, since each modelled phenomenon requires proportionately more input data. Environmental input data is a scarce commodity, not only locally but abroad too (Antonov & Kambulin, 1996).

The podium on which the simulation model is built are Rainey's (1973 & 1989) and Batten's (1967) hypotheses that major swarm movements take place downwind, towards and within zones of low level wind convergence, which will also induce

rainfall necessary for breeding. In essence, design of the simulation engine involves recording that fraction of migrants removed from an area of interest, and the prevailing wind direction assumed to have removed it. Simulations performed at other areas take into account those individuals removed from adjacent districts if they were regarded as lying upwind from the area currently in question.

Various mathematical tools - such as probability distribution functions - allow a moderately realistic spread of quantities that arise from discrete sample intervals (Davies & Hicks, 1975; Stremmler, 1982).

2.3 RELATED MODEL BUILDING EFFORTS

Hendrie *et al* (1985), when approaching the modelling problem of aphid migration, compartmentalised the model building process into three broad sub-studies:

- the environmental sensing, from whence model data are derived,
- the operational model, which integrates the data,
- intermediate and final products, including test results and forecasting.

Three fundamental aspects should be addressed when modelling insect pests in agro-ecosystems according to Hendrie *et al*. These are population dynamics, overseasoning [diapause, aestivation], and dispersion patterns. Migration refers to mass translocation of [aphids] from permanent source regions to hosts in temporarily or permanently colonisable regions (referred to by Hendrie *et al* (1985) as sinks, although more correctly, a region is only a sink if there is a net inflow of individuals). The movement is the consequence of innate behaviour. It is both initiated and terminated by environmentally induced physiological changes. The concept can be extended to include back-migration, the mass-translocation of [aphids] from temporary source regions back to permanently colonisable regions. The model proposed by Hendrie *et al* (*1985*) does not include pre-emigration and post-emigration population dynamics except as far as they influence take-off and settling behaviour.

Pivotal to obtaining accurate analyses from the wind transport model are an accurate 4-dimensional (3 space vectors and one time vector) depiction of the wind fields and an understanding of the temporal and aerial distribution of the immigrant insects. Compare this to the simplified illustration of Drake, Gatehouse & Farrow (1995), shown modified in Figure 2.2, which illustrates only two spatial dimensions.

In conceptualising the movement of aphids, Hendrie *et al* (1985) assume the division of the troposphere into four zones:

- Aphid pool on plants
- The region just above the crop canopy, the surface boundary layer (SBL), usually the lowest 10m-20m above the plant canopy
- The planetary boundary layer (PBL) - the layer of air in which turbulence and surface effects are dominant, typically 1km deep. Under condition of aphid migration, this is frequently capped by an inversion layer It is the layer of horizontal long distance movement

- The free atmosphere layer (FAL), the insect pool involuntarily uplifted by convection.

Hendrie *et al* (1985) suggest a model that allows only voluntary take-off. This means that aphids in the plant canopy can ascend only to the SBL or PBL. The FAL can receive aphids from either the PBL or SBL by convective uplift, but not by take-off from the plant canopy layer.

Aphid landing is more complex and could result from behaviourally motivated, or induced (rained out, or forced by convective downdraughts). Hendrie accommodates both, but suggests that most landings are behaviourally motivated.

The SBL contains aphids that are considered to be in a short duration mode of movement. They are not physiologically programmed for long-distance flight and will not voluntarily move into the PBL. Aphids will however voluntarily land, or they could be transported upwards to the FAL by convective forces. But if an intense convective situation arises (a squall line) the aphids would probably attempt to land - as observed in locust flight in Africa (Rainey 1973).

The PBL is considered the zone of major horizontal translation.

The Free Air Layer: Hendrie's model takes into account voluntary transport during ascent and descent. Also involuntary ascent from lower levels when convective currents produce vertical winds with velocity in excess of the flight speed. In such a situation meteorological models of air movement adequately predict aphid movement. Hendrie *et al* do not consider the possibility of involuntary descent due to physical rain-out because all of the evidence refutes such a situation. They found that National Atmospheric Deposition Program records of objects observed in wet deposition collectors - open only during precipitation events - state few cases of any insects.

WIND TRANSPORT MODELLING

In the past techniques that have been used to characterise the motion of biota through the description of air movements have included the use of wind roses, climatologically prevailing winds at one or more points, streamlines and trajectories at various elevations. In many cases incorrect assumptions have been made in the use of these techniques for interpreting migratory pathways, which leads to a divergence between estimated migratory tracks and the actual air trajectory in which the insects moved. Wind roses and prevailing winds lead to large discrepancies because these data represent long-term averages or integrations at a given locality. Migrations are usually short term discrete events occurring in response to certain meteorological stimuli (Hendrie *et al*, 1985).

Streamlines - the instantaneous directions and patterns of air movement - have been used to define the migratory path. However because migration operates over a time interval of several hours, and because the weather system causing the particular direction of air flow at any given instant moves with time, trajectories - the actual path of air as a function of time - should rather be used. Under steady state condition trajectories are equivalent to streamlines.

Yet trajectories must be used with caution too. Their magnitude and directional characteristics can change rapidly with increasing height above the surface. This implies that the height at which insect flight takes place is important so that trajectories for that height can be calculated.

Usually flight speed is much less than wind speed, especially during periods favouring long-distance migrations (Rainey, 1989, see figure 2.3 earlier), hence weather conditions, and especially air movements are the dominant factors determining migratory tracks (Huff 1963, Pienkowski & Medler 1964, Peterson 1968, in Hendrie *et al*, 1985). Hendrie *et al* also outline the problem of obtaining accurate weather measurements from a 12-hourly set of upper atmosphere soundings taken on an irregular grid of widely spaced monitoring sites.

Trajectory models must be:

- responsive to surface influences upon air movement
- able to provide interpolations of air movement between times, places and heights of measurement
- able to provide trajectories at any level where insects may be flying
- capable of including insect behavioural characteristics that might influence flight patterns.

Achtemeier (1979) in Hendrie *et al* (1985) has developed an interpolation routine based on a predictor/corrector algorithm. This interpolates trajectories at 3-hourly intervals at seven heights based on the 12-hourly soundings from the National Weather Service of the USA. Hendrie *et al* state that wind transport modelling should be used in conjunction with flight duration data. That is based on flight energy utilisation components, i.e., determined partly by aphid lipid reserves.

Aphids are not uniformly distributed in the atmosphere. Their occurrence is related to the vertical distribution of temperature and wind. Flight altitude strongly influences the insects transport velocity and in part the terminus of the migration. Observations of aphids in North America indicate that they travel in distinct layers during periods of prefrontal weather condition in late spring and in summer. (Hendrie *et al*, 1985)

Long distance flight cessation could result from several causes.

- It could be a physical phenomenon resulting in passive flight termination, e.g., precipitation or convective downdraughts could cause rain-out, or forced landing. This theory has been rejected by Hendrie *et al* (1985).
- A physiological response to exhaustion. I.e., fuel reserves are fully used up. Hendrie *et al* have rejected this also since in tethered flight tests, the aphids continue to fly long after the lipid fuel reserves are minimised. Further, in field capture tests, it has been found that aphids continue to fly throughout the night even if lipid reserves are minimised. It is thought that lipid fuel reserve minimisation may cause the insects to respond differently to environmental stimuli.

- Cessation could result from a complex combination of reactions to various environmental stimuli. This multiple cause case is favoured by Hendrie *et al*, and if it holds true then long distance, and local flight cessation may result from similar kinds of stimuli or cues.

DIRECTION OF MIGRATORY MOVEMENT AND INSECT ORIENTATION

Riley and Reynolds (1986) indicate that there is common orientation found among larger (30mg···2g) insects involved in night migrations. This phenomenon is apparently independent of the presence of moonlight. Yet little is known of the factors controlling mean direction, or degree of orientation. There appears to be NO notable correlation between the orientation and any obvious directional cues in studies done on *Chortoicetes terminifera* - the Australian Plague Locust.

Rainey (1951) confirms that the field observations had, at that time, failed to give any fixed and consistent orientation of flying locusts to potential orientating factors such as wind and/or sun. In winds of less-than-probable airspeed of locusts, up-wind, down-wind, and cross-wind orientations are all possibly observed. A swarm failing to maintain any persistent orientation, either circling (a form of swarm movement not infrequently seen) or even changing course sufficiently frequently at random, would also be displaced downwind. On that evidence, swarm movements in directions other than down-wind would appear likely to be of limited extent, and to occur only in seasons and areas of light winds. Rainey further intimates that some - possibly all - of the exceptions to these generalisations may well represent movements down non-prevailing winds.

Riley and Reynolds (1986) determined in studies conducted in south western Kenya that in some cases the insects [noctuid and sphingid moths as well as locusts] were orientated closely to the downwind direction [no specifics given]. However, there also exist cases of crosswind unimodal distributions (Riley and Reynolds (1986)) where the insects are aligned largely across the wind. Calculations on wind- and orientation vectors show that the downwind direction approximates 74° measured relative to the average insect orientation. This angle remained relatively constant, even in the face of changing wind directions!

Less commonly encountered are bimodal orientation distributions, or split distributions (Riley and Reynolds, 1986). The distinct modes observed here arise from different congregations of insects at different altitudes. The difference in mean orientation is at most 30°, and it is also seen that the deviations about the mean are sufficient to encompass a large common area. Hence the current author feels that such bimodal distributions - especially considering their rarity - are not cause for special concern.

At high wind speeds (>6ms^{-1}) migrants tend to orientate themselves within a few degrees of the downwind direction (Riley and Reynolds (1979), in Drake & Farrow, 1988). They observed that only approximately 25% of flight orientations were within 10° of the downwind bearing, yet the majority of fliers were confined to an arc ranging from 25° - 50° centred on the wind bearing. There appears to be no

evidence that they ever attempt to fly upwind in winds with velocity higher than their flight speed - typically 3 - 6ms^{-1}. The distance covered is a function of both flight duration and resultant velocity.

In work done in Kenya, Brown (1970) found a convincing relationship between the track (the path followed by a flying insect, distinct from its orientation) and the direction of the wind. The wind direction was determined and continually updated by the release of a small quantity of smoke. The direction of flight displacement of each observed insect was classified into one of 16 sectors, each spanning 22½°. The results of Brown's observations are recapped below:

- 69,5% of all insects observed displayed flight displacement directions within ±11¼° of the downwind direction.
- 90,5% of all observed insects displayed displacement directions within ±33,75° of the downwind direction.

These observations were made within 8m of the ground surface. Brown estimated maximum ground referenced speeds of 14km.h^{-1} at times when the wind speed of the order of 7km.h^{-1} were recorded, indicating that the locusts speed is its flight speed added to the wind speed. However, others as well as Brown, have determined that [in the case of the of the desert locust, *S. gregaria*] although the general direction of displacement of complete individual swarms is downwind, the orientation of groups of individuals within the swarm is constantly changing. At different times of day they may be heading down-wind, up-wind or across the wind. The net ground speed of the swarm is usually less than individuals over a long distance, since flight is intermittent and not all individuals are participating at the same time (Rainey, 1970).

Rainey (1989) quotes observations of 49 swarm observations made in Kenya, Somalia and Tanzania during the period 1951 to 1957. Here the swarm displacements showed an RMS (root mean square) difference of only 15° from the corresponding downwind directions, and in half the cases, the difference was less that 10°. In only two cases did the angular difference exceed 30°, and in both of these the swarm was unusually high at more than 900m above ground.

Taylor's (1979) model is also based on the assumption that the locusts have no preferred direction of flight and that displacement of the swarm is achieved by displacement of the supporting medium.

Scott and Achtemeier (1987) have noted that previous studies done on past migrations have produced estimates of migratory pathways by backtracking, based on data from standard meteorological observations during the time of flight. However, the problem associated with those studies is that the available data is based on ground-level winds and those at the 850mB pressure surface (typically at a height of 1500m above ground). Recall that Wantanabe's LLJ model (Wantanabe, 1995) also utilised measurements at the 850mB pressure surface. These data do not adequately resolve the vertical structure of the wind, particularly in that layer between the two levels (surface and 1500m). This zone is

a region that many researchers have found to be favoured by flying insects. Hence, back-trajectories based solely on these data will provide only general estimates of migratory pathways.

Once airborne, most insects move with the airflow. Due to their small mass, all but the largest insects will have negligi y small airspeeds in relation to the wind velocity. Their horizontal transport will thus be essentially passive (Rose, 1975, in Scott and Achtemeier, 1987). Trajectory analysis is based on this assumption.

Citing studies done by Clarke in 1983 on inanimate transport using trajectory analysis to model flow of pollutants, Scott and Achtemeier (1987), outline three different numerical techniques evaluated.

The salient features of these computations were:

- There is an apparent relationship between simulation accuracy and the use of synoptic weather patterns in contrast to regional weather patterns.
- The highest accuracy of simulation was found to occur during well defined flow patterns, especially on the lagging side of high pressure systems.
- For complex meteorological conditions, increases in the data density DID NOT result in better performance of the trajectory technique.
- Meteorological events on time scales of less than six hours apparently contributed significantly to simulation error.

Insect pests have shown an ability to change height during flight, although a large number of migrants travel in the [convective] boundary layer; most likely by choice (Scott and Achtemeier, 1987). Work done by Rose in 1975 (in Scott & Achtemeier (1987), and Hendrie *et al* (1985), indicate that there are preferred regions of flight in the areas of warmest temperatures and highest wind speeds. These are layers usually associated with, and found near the convective boundary layer's upper limit. This means that knowledge of the timing and height of the insects' flight is paramount for the usefulness of trajectory analysis to be maximised.

Scott and Achtemeier (1987) did their work in North America, their data being provided by the NWS (National Weather Service). The NWS's weather stations form a roughly uniform grid over the entire continent, with an average interstation distance of 365km. The theoretical horizontal resolution of this network is twice that distance, i.e., approximately 700km. Given the widely spaced locations, and the temporal frequency of routine observations above the surface, Scott and Achtemeier question the limits of confidence that can be placed upon back-trajectories created from this data. Suspicion must then logically fall on forward trajectory accuracy also. Small-scale meteorological features that may play an important rôle in some insect migrations could not be adequately resolved. The upper air measurements were available only for 12 hour intervals, which is too coarse for accurate temporal resolving of small-scale disturbances, or for rapidly moving synoptic-scale weather features.

In developing trajectory analysis techniques, Scott and Achtemeier (1987) observed:

- Air near the ground surface blows parallel to the earth's surface, but at elevations above 2000m (above ground) the flow tends to follow surfaces of constant pressure, EXCEPT near weather fronts.
- The advantage of performing the trajectory analysis on constant pressure surfaces is that most other insect studies have relied upon existing constant pressure charts. Also, upper air meteorological observations are usually made for certain pressure levels.
- Pressure surfaces within the lower troposphere can intersect with the ground. The outcome of this phenomenon is twofold:
  a) coding problems, and
  b) the need for extrapolation of meteorological variables below ground level.

Their solution is to utilise a co-ordinate system (mapping) that ensures that lower surfaces of constant pressure follow the ground, and the upper surfaces follow a surface of constant pressure.

An accurate, objective streamline technique developed by Achtemeier (Achtemeier,1979, in Scott & Achtemeier, 1987) was used to calculate Scott and Achtemeier's back-trajectories. These trajectories are constructed from eight segments, each one for a three hour interval, thus covering a full 24 hour period preceding the observation. Bilinear interpolations are used to calculate wind vectors at points not coincident with actual points of measurement. It is assumed that for the interval of 3 hours over which the trajectory segment is constructed, the wind blows uniformly in that region.

The results obtained by Scott and Achtemeier (1987) are summarised below:

- Back trajectories for the eastern site (Jackson, Mississippi) averaged approximately 90km longer with the improved temporal resolution. At the same time, at the western station (College, Texas) endpoint differences maximised at 180km, yet varied elsewhere from 40km to 100km.
- The effects of back-trajectories calculated with wind observations using data from extra sites (where available) in the analysis network roughly paralleled the patterns calculated with only regular data.
- Increases in either temporal or spatial data (but NOT both) frequencies generated endpoint differences that were generally not greater than 150km. Larger difference were produced when both were combined but only at the western station (Texas).
- The authors state that although the results may seem to show substantially large errors, they are still considerably less than the potential errors arising from calculations omitting improved vertical resolution of the data.
- One should expect 24 hour back-trajectories calculated from the NWS upper air network (800mB pressure surface) to produce errors in the trajectory end point locations (which is in fact the migratory journey starting point) that can range from 0 to 250km, or more, depending on the persistence of wind, or lack thereof, throughout the 24 hour period.

- These errors are small in comparison to the approximately 800km separation (at the western site) between the endpoints of the surface and the 1900m trajectories, assuming in both instances that the same air passed over the western site 24 hours later.
- In a shearing environment, in early spring weather systems, uncertainties in locations of the origin of migrating insects can be decreased if back-trajectory calculations include more fully the vertical structure of the boundary layer wind field.
- An extreme event - the nocturnal jet - is a short-lived (6 - 12h) night-time wind speed maximum occurring frequently in the lower 600 - 1200m altitude under certain weather conditions. The nocturnal jet may pass through its life cycle between regular upper-air observation times. Then large errors between back-trajectories and the actual migratory paths should be expected, regardless of the methods of calculation. Because of the potential importance of this phenomenon for transporting insects, a method of incorporating it into calculations should be sought.

Rainey (1951) has said that it has been found useful to examine some of the meteorological implications of down-wind movements. A synoptic chart, recording observations of weather over a large area, demonstrates across which boundaries surface winds show a net inflow of air over outflow, and also reveals areas within which the overall motion of the wind is one of ascent. Also shown are areas of divergence, across the boundaries of which, outflow exceeds inflow, and within which, the overall vertical motion is one of descent.

Note that divergence is generally associated with fine weather, while convergence usually leads to widespread and heavy precipitation. In general, winds blowing within a few thousand feet of the surface are considered as ultimately blowing from zones of divergence to zones of convergence. Therefore, downwind displacements may be expected to carry airborne insects into regions of convergence, and swarms may be expected to aggregate in such areas.

Convergence also provides a mechanism for the frequently recorded association of exceptional rains with the arrival of locusts (Rainey, 1951). A similar effect, he states, may perhaps also contribute at times to the process of *gregarisation*, by producing a measure of real concentration of locust numbers into areas of moisture and vegetation necessary for breeding.

Brown (1970) has also made a relevant observation regarding the night migration of insects compared to those that fly during the day. Attempts to back-track immigrant insects to biologically feasible source areas, although nearly always giving plausible results for night-flying moths, has proven much less satisfactory in the case of butterflies - which fly predominantly during the day.

Since the methods used for back-tracking presuppose down-wind movement, this evidence may be regarded as suggesting that migrating butterflies can, and do, maintain an orientation other than down-wind over long periods, whereas the greater success in back-tracking moths is due to their flight orientation being down-wind or at random. The relevance of this is thus that locusts - many species

of which migrate at night, might be expected to have flight paths that closely match the downwind direction. This should be considered in addition to findings that no convincing cues - such as compass bearing or lunar azimuth - could be found for orientation guides in the case of locust flight.

Drake and Farrow (1983) performed trajectory calculations on Australian plague locusts, covering distances of the order of 200km. They assumed as necessary and acceptable that all locusts took off at dusk. This contradicts the observations of take-off being spread over a dusk-to-midnight period.

At any particular time, the speed and direction of flight at locations away from the observation site were the same as that measured at the site. This is likely to be valid (according to Drake & Farrow) as there were no major terrain features in the areas over which the locusts were believed to have flown. The upper air winds would thus have been uniform over a wide area.

Clarke's (1971) work on *C. terminifera* indicates that these insects take to migratory flights shortly after sunset. This migratory mode of flight differs from arbitrary daytime flights in that the take-off climb-rate is substantially steeper. Clarke postulates that the trigger for these flights is falling light intensity. The threshold temperature for flight has been empirically estimated to be 17,5°C. Relative humidities during these observations ranged from 9% to 70%, although there is no indication that these are limiting values. Wind speeds exceeding 1,8ms^{-1} resulted in the locust flying substantially downwind, while at lower wind speeds the flight orientation was less well defined.

This random orientation of fliers with respect to wind at low wind speeds suggests that under dry anticyclonic conditions, where pressure gradients are slight, flights after sunset will lead to widespread dispersal. Downwind flights at the higher wind speeds are associated with the passage of depressions - low-pressure troughs and thunderstorms, or cold fronts. Thus night flying locusts are not only concentrated by the passage of depressions, but are also likely to encounter rain and conditions favourable for breeding (Drake & Farrow, 1988; Pedgley, 1979).

The wind speeds do not appear to be a decisive factor in determining whether or not takeoff will occur. Clarke (1971) has observed takeoff in winds with speeds ranging from 0,5 - 6,5ms^{-1}. Clarke maintains that at the time of writing (1971) nothing was known concerning the sensory input which induces locusts to land where they do. At collection lights, large numbers often appear when there is a sudden change in wind direction or speed. Cessation of flight is not likely to be caused by an abundance of available food either, since large masses of individuals are known to land on ground devoid of vegetation.

Kisimoto and Sogawa (1995) - in studying long-winged planthoppers in Asia - have noted that these insects exhibit both dawn and dusk takeoff during summer, but during autumn the dawn takeoff is much reduced. Further, the dusk takeoff is reduced in windy conditions, and is completely inhibited when the near-surface windspeed exceeds approximately 3m.s.$^{-1}$ (6 knots). This is a relatively light wind, but it must be kept in mind that this refers to surface measurements (10m altitude) and higher altitude winds will have substantially higher velocities. Investigations into the cessation of flight lead to speculation that landing of planthoppers is

associated with rainfall events, frequently combined with cold fronts. Turbulence at the front is suspected of suppressing active flight and forcing insects to land. This is substantiated by laboratory tests in which insects subjected to airflows of greater than 5,5ms^{-1} (11knots) cease wing-beating. The destination locations of migrations were frequently found to be inhospitable to the planthoppers, suggesting that migrations may end almost randomly if appropriate flight conditions are not maintained.

OTHER FACTORS INFLUENCING POPULATION REDISTRIBUTION

Australian Plague Locusts caught under lights were examined by Clarke (1971). It was found that in most cases the specimens' alimentary tracts contained either no food or very few plant fragments. The examined individuals were also found to range in sexual maturity from entirely immature to advanced stages of maturity. A small proportion were ready to lay with fully formed eggs in the oviducts. There was also evidence that some of the females collected had recently laid and were beginning another cycle of reproduction.

Wright (1981) in her treatise on the origin and development of plagues concludes that her data show no obvious pattern leading to plague generation in the Australian interior. Neither do the overall monthly or seasonal rainfall averages seem to be of major importance. This claim is supported by the fact that plagues have occurred following wet winters and heavy spring rain - such as in 1978 - or after light winter rain and dry springs - as in 1972. This concurs with the observations of Price (1988). However, the first major rain of the Australian locust season generally occurs in October, usually following a dry winter.

Wright (1981) determined that for each of the four Australian plagues studied, a breeding sequence leading to the peak infestation can be followed over either three or four consecutive generations in the Australian interior. In each generation major rain fell either after peak fledging - causing the adults to mature and lay locally - or before peak fledging - producing adults which migrated. Rainfall in the Australian interior varies greatly between years and also geographically within any season. This means that the average rainfall is useful as a very general guide only. Outbreaks decline if rain fails to fall at any of the critical stages in the plague generation pathways. Conversely, sequences of very heavy rain can cause the majority of adults to lay locally in each generation. The rapid onset of egg maturation caused by such rainfall only allows two or three days for migration and this period will not necessarily coincide with suitable migratory conditions.

Further, while immigrants are able to lay in dry soil, rainfall around the time of an invasion may make the difference between the decline of an outbreak and the development of a plague. Hunter (1981) claims that mass take-off requires that locusts with adequate fat reserves must experience windy conditions that would normally inhibit daytime flight. Daytime flight can also be inhibited when it is exceptionally hot (>37°C). High surface winds and high temperatures often occur ahead of front and troughs, which might explain why migration is associated with the approach of such systems. It has been suggested that locusts are seeking the rain which these systems often cause, in order to lay. Immigrant locusts are known

to lay without rain, although persisting (resident) locusts with little fat will not lay. Hence, laying in dry soil after migration is probably the general rule in low rainfall areas.

The power for flight is derived from the oxidation of stored fat, and an upper limit for the potential duration of continuous flight without feeding is set by the ratio of size of fat resources to the rate of consumption in steady flight. Weis-Fogh (1952) in Taylor (1979) estimated a maximum endurance of locusts flying in a wind tunnel to be around 20 hours of continuous flight. Taylor suggests that these estimates require some revision - based on Wooten and Sawyer's (1954) observations. The distribution of individual's flight endurance is of extreme importance to Taylor's model, and at that time (1979) was unknown.

Taylor's (1979) model assumes that the flight time is determined solely by the ratio of an insects' fat reserves to the rate of fat consumption. A log-normal probability distribution is applied to this state variable, with parameters such that the median flight time is 10h, with approximately 6% of individuals achieving a flight time of up to 60h. Taylor suggests that the log-normal distribution is preferable to the Gaussian distribution since the latter exhibits a symmetrical distribution about the mean. This, he claims, is biologically improbable. Note that in the current author's model, the Gaussian distribution is NOT applied to a biological phenomenon, but to air (fluid) dynamics, where the Gaussian distribution's symmetry more closely relates to expected outcomes.

The current simulation model does not incorporate any interpretation of the insects fat reserves or indeed any energy balance equations. At this time there is no environmental data available which can give any indication of the fat reserves, or consumed vegetation.

INTEGRATING OBSERVED PATTERNS INTO A SIMULATION MODEL

The following chapter "The Modelling Approach", describes some of the characteristics of classical and contemporary models. The developmental history of the locust lifecycle model is reviewed with reference to commonly encountered model building techniques, and some special features that have been included into the model. A detailed mathematical description of the current model is provided with the emphasis on the most recent developmental phase, namely wind transport modelling.

Chapter 3 THE MODELLING APPROACH

3.1 MATHEMATICAL MODELS DESCRIBED

It is convenient to conceptualise a model as a representation of an existing structure, on a smaller scale. This often involves a simplified (always mathematical, according to Worner, 1991) description of a system etc., to assist calculations and predictions.

Starfield, Smith & Bleloch (1994) define a model as being "a purposeful representation". They suggest that the model does not have to be a complete representation, and should only include those features of reality that are essential to understanding the entity under consideration. Removal of spurious detail allows the model to be more easily understood and better controlled. When designing computer models, smaller more concise models are desirable (McKinion, 1992) since they require less in the way of computer resources (computing power and hardware) and associated software will frequently execute faster. This is in addition to the universal advantages of easy maintainance and understanding. Any model is only an approximation to reality. The key to successful modelling lies in striking an optimal balance between the accuracy of the approximation, and the complexity of the model.

In the case of a time variant phenomenon, it is necessary to represent the physical characteristics of the system, and then also to emulate how these change as time progresses. In this way, a static model has become a simulation. The term "simulation" has been formally defined by Maisel & Gnugnoli (1972, in McKinnion, 1992) as "a numerical technique for conducting experiments ... which involves certain types of mathematical and logical models that describe the behaviour of ... biological [and] physical systems - or some component thereof - over time". Maisel & Gnugnoli erroneously imply that simulation modelling can only be performed using digital computers; there is nothing inherent in the simulation process that confines it to this type of machinery. McKinnion concludes that, for complex systems, the use of simulation methodology is unequalled in its ability to achieve realistic models of system behaviour at a reasonable investment of time and money.

In referring to a system, it is important to define and understand that system's boundaries. Those entities within the boundaries comprise the system itself, and those without comprise the environment. Systems which perform no interaction with their environment are closed systems. In practice, all biological systems are open systems, and the simulation model therefore requires a link to the environment. In the locust simulation model, this link is achieved by the input data set, which comprises detailed weather records

The concept of a 'dynamic simulation model' arises when the model of the physical process becomes self-configuring in response to simulation conditions. In contrast to a static representation of the process - which is unchanging in its specification throughout the simulation, the dynamic simulation model has the capability to adapt its structure in response to the conditions it is simulating. This is

not to say that the software code re-writes itself, but different code can be selected from a library of available functions. The functions most appropriate to the prevailing conditions can be used, thus creating a suite of many different models, where only the best is utilised under any given circumstances.

Formal definitions of models distinguish two sub-species: the linear algebraic model, and the rule-base model.

A linear algebraic model makes the assumption that the processes in models are usually mathematically smooth. In other words, the phenomena can be measured to a theoretically infinitesimal degree of accuracy. In certain instances there are exceptions to the smoothness of the functions and discontinuities or singularities do arise. These models frequently make use of linear (or more rarely, non-linear) differential or difference equations. In the case of the difference equation, the smoothness of one of the model parameters has been sacrificed by considering the quantity to increment in discrete steps. A typical example is time. In some cases it may be more convenient to consider time in discrete quantities - such as days, or years - instead of a quantity that can be regarded as continuous with no lower limit (in theory) in the fineness of its measurement.

Non-linear differential or difference equations are often exceptionally difficult to solve, and can often only be solved by regarding a limited range where the equations can be approximated by linearised forms.

The term non-linear specifically refers to non-linearity in the context of the state variables of the system. (A formal definition of state variables is included in the section "How the lifecycle simulation operates II", later in this chapter, where it is more appropriate.) If the environmental variables are assumed to be periodic (as is the case here) then the environmental input can be regarded as a periodic forcing function. The dynamics of a forced system which is linear in the system variables is simply a stable system which is "pushed around" by the environment, no matter how complicated the forcing function is (Allen, 1992). Stability implies that the system decays to some finite steady state. While environmental forcing can result in some highly complicated and surprising ways with non-linear systems, the outcome with linear systems is basically simple. Most insect lifecycle models to date are of the linear type with complicated environmental forcing functions (Allen, 1992), and the non-linear rule-base approach of Nailand and Hanrahan (1993) is unconventional.

The rule-base model differs from the linear (or non-linear) algebraic model in that it does not use continuous mathematical relationships to describe cause/effect phenomena. Broadly speaking, it can be regarded as a sequence of if/then decision interrogations. This lends itself to representation by a flow diagram. At key decision points, a question is asked. The model is interrogated concerning prevailing conditions, and then decisions are made as to the course of action to be taken in response to those conditions. A data base exists; this contains the rules by which the decision engine must abide.

The attractiveness of the rule-based model lies in the fact that it is not necessary - or even desirable - to develop mathematical relationships to describe the

processes. If the cause/effect mappings are known, then the model grows directly from them.

These two model types describe extremes, and it is not unusual for a model to incorporate components of both. Such a hybrid is known as a rule-enhanced model (Starfield, Smith & Bleloch, 1994).

It is not always a characteristic of a simulation that it must execute faster than real-time (in instances when the process is too fast for observation it might be necessary to model it in order to monitor it at an acceptable rate). Yet in our case, it is extremely beneficial for the simulation to execute as rapidly as possible. This is to facilitate as many simulations as possible in the available time. A large number of simulations will allow a more objective evaluation of the model's accuracy. Many of the physical characteristics of the model are known only vaguely, and it is by such simulations that they can be fine tuned. The essence of this approach is that a great many successive simulations may be executed, altering the physical characteristics between each execution and the next. By observing how the results converge towards, or diverge from the actual recorded results, it is possible to estimate more accurately the optimum value of the changed parameter. It also provides information on the sensitivity of the phenomenon with respect to that parameter. A more detailed discussion of parameter estimation and sensitivity analysis is provided in chapter 4.

Once the model has been defined, a key question needs to be asked: 'How good does it have to be?'. This question might initially seem paradoxical, but it is not necessarily true that the most accurate model is the best model. A highly accurate model will frequently incorporate unnecessary details, with two undesirable consequences: firstly, the model becomes difficult to understand and maintain, secondly, simulation execution time is poor.

The following tentative answers to the above questions are proposed:

- If it is fast, it is good. If it is slow it is bad.
- If it meets the required specification for accuracy, it is good enough.

It is important to note that the criteria for a good model are highly dependent on the application, and no general rules should be applied, Not all situations, for example, demand rapid execution time.

In this application, there is nothing to be gained from building a model that is too complex or detailed. As described in later sections, there is insufficient testing information to verify the accuracy of the increased resolution. Furthermore, unnecessary detail will cloud understanding of the basic functions.

STOCHASTIC AND DETERMINISTIC MODELS

Models, whether they be linear algebraic, rule-based or rule enhanced, may also be sub-divided into the categories of stochastic or deterministic. Deterministic models are entirely predictable in their behaviour. Stochastic models introduce the element of probability, and of trying to anticipate the likelihood of an event. The classic "Schrødinger's cat" thought experiment relating to quantum physics is an interesting example: the conclusion drawn from there being an equal

probability of the cat being alive or dead is that the cat is simultaneously alive and dead!

Insect population systems are always embedded in the larger physical and biological environment, and this is reflected in their equilibria and dynamics. These systems are highly dependent on how and where the system boundaries are defined. Allen (1992) claims that, even with a meticulous degree of care, isolation of the system is never complete, and this failure to confine all links to the environment to known, defined channels manifests itself as random or pseudorandom perturbations in the model, i.e., noise. Since noise represents an unpredicable component, it is conventionally described as random error. This interpretation of apparent randomness in an otherwise simple deterministic model is comparitively new, according to Allen. Recently, Higgins *et al* (1997) have shown that even relatively small perturbations, uncorrelated from year-to-year, can lead to large population fluctuations - with a multi-year period - in deterministic biological systems.

Generally, if the number of simulation runs is large, then the average of the results given by the stochastic model will tend towards those provided by the deterministic model. The stochastic model will, however, provide an indication of the variability of the outcomes from the mean, and also the confidence we can have in the simulated result.

Starfield *et al* (1994) show that if chance is not an important part of the phenomenon, then there is nothing to gain from building a stochastic model. The stochastic details only serve to confuse the understanding of the problem.

The locust simulation model relies on climate data, and the simulation results are evaluated upon actual locust count data. It is known that neither of these data sets is particularly accurate. Furthermore, the level of accuracy is unknown. It is therefore difficult to estimate a confidence interval on any of the simulation results. For this reason it was decided that no additional random elements should be introduced. The concept of a probability index is introduced when considering the wind data, but this does not make the model stochastic. There is no randomness associated with these probability density functions; all are based on unvarying mathematical formulæ, and are thus predictable and the experiments highly repeatable.

3.2 A HISTORY OF THE SIMULATION MODEL'S DEVELOPMENT

A FIRST APPROACH TO DEVELOPING A MODEL

In 1993, P. Nailand and S. A. Hanrahan published the results of their development of a population model for the Brown Locust (Nailand & Hanrahan, 1993). That model, coded by P. Nailand, for Watford Pascal™, and running under IBM's VM/XP operating system required (relatively) large amounts of computer memory and relied on the University's mainframe computer - which was subsequently decommissioned. The model was converted by P. Hanrahan to run under Mathworks™ Matlab™ ver. 4.2 with Simulink™ graphical support. Matlab is available for Microsoft Windows™, and this operating system environment was

selected due to its universal appeal to PC™ users. Although Matlab is available for the more powerful UNIX™ operating system, the PC was retained as a hardware platform in order maintain transportability within the Zoology Department.

Matlab was chosen as a programming language because it was available to the Zoology Department, and because it allowed for great flexibility, compact model structure and efficiency in model design. Matlab is specifically engineered to cope with operations on exceptionally large matrices. Additionally, matrix operations are an integral part of the command set, facilitating easy programming and rapid execution. At the same time as the conversion to Matlab, the original Watford Pascal source code was translated to Borland Pascal™ - which is supported by the PC operating system environment. Although the code executed, it was found that the Borland compiler placed severe restrictions on the maximum dimensions of variables within the program. The Pascal approach was then abandoned in favour of Matlab - a more versatile interpreter.

Nailand & Hanrahan' s (1993) model was rule-based. Parameters of the locust life-cycle were used to generate a set of rules according to which population numbers changed in response to changing rainfall and temperature conditions.

So too was the first-generation Matlab model rule-based, but during its subsequent improvement it has grown to incorporate various characteristic components of the linear algebraic type of model, and so is best now described as a rule-enhanced model.

The first generation Matlab model became the starting point for the ongoing development of a migration model. The specific aims of the revised model have been outlined in Chapter 1 - *Outline and objectives*.

Problems encountered with the earliest Matlab model were:

- The model operated only at a single geographic location, on a single grid co-ordinate generating changes in locust population numbers at that point in space. No account could be taken of portions of the population being removed or added due to phenomena such as migration.
- The technique used for storage of locust and egg numbers did not lend itself to rapid index addressing of the matrices. The simulations were stupendously slow.
- The matrix structures used were unnecessarily large, generating variables in the order of 60MB in size - too large for the available harware to deal with.
- Errors were discovered in the formulæ used to compute degree-days.
- The output of the simulation failed to monitor stagnant (or steady state) populations. Only transitions between stages were recorded, and populations that remained constant in size were not recognised.

- The simulation retained records of all individuals that had died. Consequently large areas of storage space were being wasted in tracking dead individuals.

In the light of these errors, it was concluded that the results from the earliest Matlab model were of little value, although the basic model structure was sound. Later revisions relied heavily on the initial structural layout provided by that model.

The structure of the model is derived from the locust lifecycle block diagram, formulated by Nailand and Hanrahan (1993) and revised by P. Hanrahan (1994, pers. comm.). This visualisation has been retained from Nailand's original model, although the Nailand model was not based on a graphical user interface. The locust lifecycle is well documented in various texts (Price, 1988; Matthèe, 1951; Matthèe, 1961; Lea, 1970).

Mathworks Simulink™ is a companion application to Matlab, which allows realisation of block diagram structures within a model. The customary (classical) technique for building such a simulation model is to represent each transition by means of a difference equation (or differential equation, in the case of continuous time analyses). When considering any lifecycle stage, *instars*, for example, the population of instars at time *t+1*, $P_{(t+1)}$ becomes some mathematical function, F, of the population at time *t*.

$$P_{(t+1)} = F\{P_{(t)}\}$$

It is not too difficult to convince oneself that *any* physical (real) process can be described by an nth-order differential equation (d.e.). The more complicated the process, the higher the order of the equation. Using matrix algebra, a *linear* n^{th}-order differential equation can be decomposed into n first-order differential equations, which are substantially easier to solve than the original n^{th}-order equation. This decomposition cannot be performed with non-linear systems, another reason why they are so difficult to deal with.

The time increment, *t*, can be chosen to be any value that provides sufficient resolution. In the case of the locust model, a time step of one day is convenient since weather records are available to a resolution of one day, and the entire life cycle is completed in a matter of weeks. Events do not occur rapidly enough in the lifecycle to merit using finer timeslices. Using the largest possible timeslice is desirable, since the size of the data structures that record the lifecycle history are inversely proportional to the length of the timeslice. Shorter timeslices require more storage space for a given simulation period. The need for timeslices as short as one day is perhaps debatable; modelling lifecycle events themselves does not require such frequent updates, but the nature of Karoo weather dictates that each day must be simulated so that rainfall - which might occur only on a single day in a long span of dry conditions - is not overlooked (Price, 1996, Pers. Comm.).

Simulink allows the programmer to paste the block diagram directly into the model. The block diagram is then not an abstraction of the model, or *vice-versa*. On the contrary, the block diagram actually *becomes* the model. Each block within the block diagram encapsulates the appropriate representation of the

applicable difference equations, and the visual link lines between blocks actually carry information between relevant blocks, implicitly. The programmer is not compelled to program channels of information interchange between blocks; this happens automatically when the blocks are linked graphically. This model building technique is inextricably linked with object orientated programming structure (OOPS), a concept that will be described in more detail later.

DEVELOPMENTS WITHIN THE CONTEXT OF THE MATLAB DYNAMIC SIMULATION MODEL

Revisions of the simulation model, for which the current author is responsible are: streamlining of the program code, exploiting the capabilities of Matlab in the model design and correction of computational errors discovered.

Additionally, the following progress has been made:

- The most substantial advance has been that the model has been extended to incorporate wind data and to allow immigration and emigration of locusts between stations. The model at present deals with two stations at one time but can be extended to deal with any number of stations when the need arises.

  Understandably, as more sites are added to a simulation, so more demands are made upon the computer's resources. Simulation models incorporating many sites will require both fast processors (high speed Pentium™) and adequate memory (upwards of 16MB RAM)

- Spreadsheet capabilities have been added. The input to the simulation (weather data, and actual locust count data) is now presented to the model in the form of a Lotus 1-2-3™ (or compatible WK1 format) spreadsheet file. This allows many database applications such a Lotus™, Quattro-pro™, D-Base™, Q&A™, etc. to generate climate data for the simulations. There is no need - as was the case before - to convert data into a specialised model-compatible format.

  Furthermore, the simulation model output is now also available in Lotus WK1 spreadsheet format. This allows users the option of loading the simulation results into their database application of choice, and using any of the analysis tools which that application offers (graphs, histograms, regressions etc.).

- The simulation model incorporates some simple statistical analysis tools. [illegible] correlation tests have been chosen as the most useful tests, in terms [illegible] type of data available from the simulation. Various such tests may [illegible] be performed within the simulation, without the need to export data to a statistical analysis application. The details of these test are discussed in Chapter 4, under the section "A Statistical Approach".

- The simulation runs substantially faster, due to improved techniques for assigning variables within the program code. Typically a two-year

simulation, running at two sites simultaneously, can now be accomplished in around five minutes (80486-DX4 99MHz CPU with 16MB RAM).

Such a simulation generates workspace variables that require under 2MB of RAM. However, Microsoft Windows and Matlab's command interpreter demand that the machine has a minimum of 8MB of RAM, for reliable operation. 16MB is preferred for stable Windows multitasking (other utilities running in the background). The salient feature here is that adding one additional site, for instance, will not require an additional 64MB of memory (virtual or hardware), as was the case with the earlier version of the simulation model. In fact, 16MB of RAM is speculated to be adequate for simulations involving up to six sites.

THE LIFECYCLE MODEL

The lifecycle model is heavily based on the work of Nailand and Hanrahan (1993) and Hanrahan & Hanrahan (1994). Rules for the lifecycle stage transitions have been retained from Hanrahan & Hanrahan (1994), with minor modification (Price, pers. comm. 1996). Dramatic changes to the model have been in the way that the source code represents various populations. The matrices, or array variables, created within the simulation have been completely redesigned to optimise speed of access, and required storage space (RAM).

As stated earlier, the lifecycle model remains an independent entity - even within the context of the migratory simulation model. The model retains a hierarchical structure. The top level of the hierarchy represents the broadest description of the model, and the nested structures are added relating finer and finer detail as one progresses down the hierarchy, and as more accuracy is required.

Such a hierarchical structure is shown below in figure 3.1.

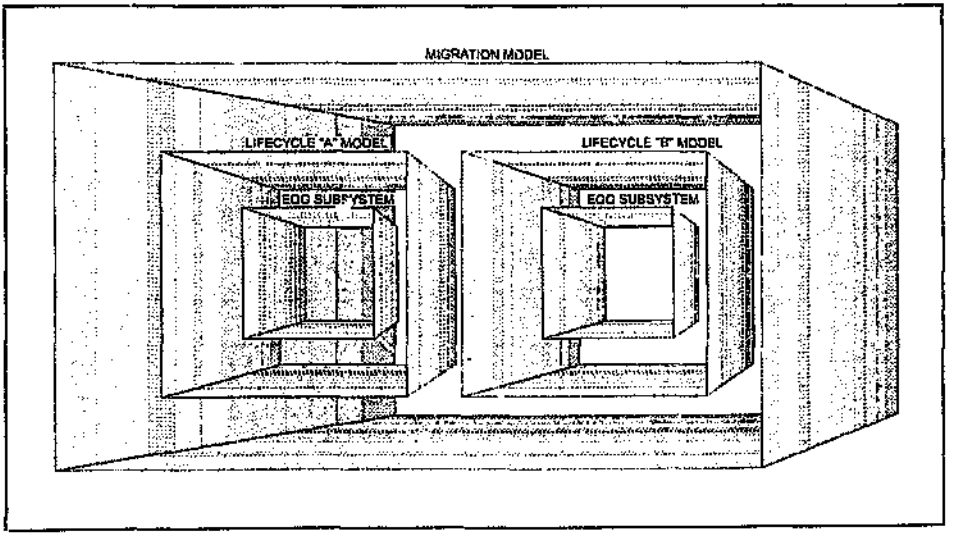


Fig. 3.1 Nested hierarchical model structure.

OBJECT ORIENTATED PROGRAMMING SYSTEMS

Object Orientated Program Systems (OOPS) is a revolutionary programming methodology that has rapidly become the programmers' buzzword of the nineties. It provides a powerful model development and building tool, and its adoption is encouraged by the inherent design of Matlab and Simulink. McKinnion's 1992 formal defintion of OOPS is one in which a model of a system can be viewed as a collection of system components that are treated as objects. These objects have knowledge within them concerning their function. Objects are activated by messages (information) from other objects, and the other objects do not need to know anything about the object to which the message is passed. The simulation model is heavily based on OOPS methodology. The primary advantage of OOPS programming is maintainability of software code for very large systems. McKinion proposes that 100 000 lines of code is the maintainable limit for conventional structured programming approaches, whereas OOPS methodology imposes no such limitations.

The simulation model in its present form consists of slightly more than 1MB of source code. The actual number of code lines has not been counted, but it is estimated to be between 25 000 and 50 000. This is not yet beyond McKinion's theoretical limit. Notwithstanding, the OOPS approach facilitates faster, more accurate program development and teaching, even for models that can be coded in a few hundred lines. More importantly, OOPS allows the model to be customised or upgraded very easily, simply by replacing the obsolete object.

BLACK BOXES, INTERFACES AND PORTS

The top level of the hierarchy is represented by the inter-lifecycle interactions (Figure 3.1). At this level, the lifecycles are considered closed entities (objects), or "black-boxes". No clues are visible at this level as to what goes on inside these compartments, hence the term black-box. Each lifecycle black-box represents the lifecycle of a particular population of locusts, at various stages of development, at a single geographic location. Each lifecycle entity (object) has a well-defined interface, by which it communicates with the environment, and with other such lifecycle entities. This interface comprises two sets of variables - and possibly constants - making up both an input interface and an output interface, conveniently referred to as ports. The input port conveys information required by the lifecycle system in order to perform its pre-requisite activities. The locust lifecycle parameters constitute a component of the input port. The results of the computations performed within the lifecycle black-box are available at the output port. (see Figure 3.2 overleaf)

This design, incorporating sub-systems with invisible inner structure, is very useful in creating an ultimately complex model. At each stage there is no unnecessary detail visible to cloud the major issues under consideration. At the migratory level, it is not useful to know what kinds of processes are occurring within the various developmental stages. All that is required is knowledge of the number of flying adults. If this information is available at the output port of the lifecycle black-box, then the model is perfectly adequate *at this level*. How this

information is generated at the output port is of no concern - yet. That is a problem to be considered only at the next step down the hierarchy.

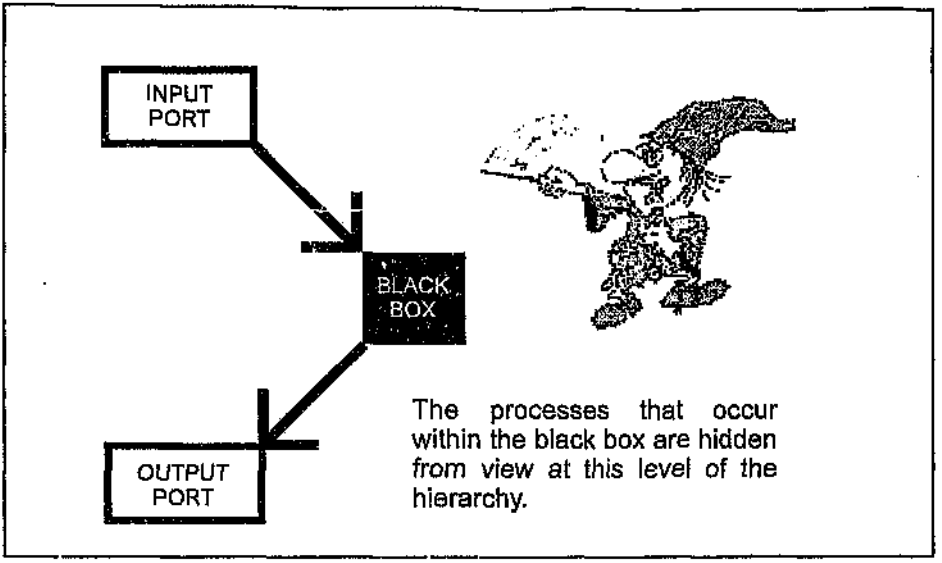


Fig. 3.2 Black box diagrammatic representation.

This description of how detail is hidden, and made accessible only to levels which require it, is used throughout the simulation model. This must be remembered when considering any level of detail within the model.

THE LOCUST LIFECYCLE

The Brown Locust lifecycle is complex, particularly when considering egg stages. This is one of the reasons it is so good at surviving the inhospitable conditions of the Karoo region. The model, although developed specifically to model the Brown Locust, has no inherent characteristics that make it exclusive to this species.

Locusts are grasshoppers, distinguished by their pronounced, density-dependent polymorphism. About a dozen species in the short-horned grasshopper family, Acrididae, have evolved this capacity to change their habits and appearance according to their population density. When populations are sparse, the individuals live separate lives and migrate singly, often at night. As their numbers increase, however, they respond to more frequent encounters with one another by becoming more and more gregarious, active, and conspicuous. They form bands of hoppers (nymphs) and adults, which migrate by day in a uniquely itinerant mode of life, mating and reproducing as they travel. The swarming and non-swarming forms are known as the gregarious and solitarious phases. So different are these in appearance, that they were not recognised to be of the same species until 1923 (Faure, 1923).

The solitary phase is always present, whereas the gregarious phase appears only sporadically. Under sufficiently high population densities, a solitary hopper will mature into a gregarious adult, with longer wings, shorter legs, and other identifiable characteristics, most obviously changes in colouration, but also behaviour (Backlund, 1955; Pick & Lea, 1970). As more and more of the gregarious types appear, migratory swarms build up. Once started, a locust plague may last for several months - or even years - while generations of the swarms overrun regions in which the solitarious phase cannot maintain itself. The swarms eventually die out and scatter, leaving only the solitary phase.

The lifecycle model distinguishes gregarious and solitary forms of adults. But it does not automatically determine whether nymphs will progress to gregarious or solitary adults. The phase ratio at present is a fixed quantity, adjustable by the user. Further developments might allow the model to continually update the phase ratio depending on the instantaneous population density. However, Wolda and Dennis (1993) suggest that density dependant mechanism tests only become statistically significant after 20 to 30 generations, and the short period simulations performed thus far are unlikely to stimulate these mechanisms.

A superficial description of the lifecycle follows, however, if concise numeric details are required, the reader is referred to the accompanying technical reference, which describes the rulebase in detail (Appendix F). Inspection of the rulebase reveals how these lifecycle characteristics are incorporated into the model. The lifecycle block diagram, figure 3.3 overleaf, will also be helpful in understanding this description, as will figure E5.1 in appendix E.

The female locust lays eggs in sandy soft soil, typically 40 to 80 at a time, and embeds them in froth as an egg pod, about fifty millimetres beneath the soil surface. These eggs may be diapause or non-diapause, and both types may occur within a single pod. Waterlogging is fatal to the eggs, but the embryos have to take in a certain amount of water or their development stops. Sufficient hydration of the embryo causes it to develop to a stage known as catatrepsis, beyond which it can no longer return to a state of dormancy. If conditions become unfavourable again (too dry), eggs beyond catatrepsis will die. Normally not all eggs in an eggbed will hatch together. Approximately five percent of these eggs - those laid as diapause eggs - remain in a state of dormancy, or quiescence. Non-diapause eggs may also enter a state of quiescence if temperatures remain low for long. Quiescent eggs may be resurrected by sufficient moisture and prolonged warmth, otherwise they can - if conditions are not too severe - remain viable in quiescence for up to three years (Matthèe, 1951; Price, 1988).

The newly hatched larvæ quickly moult to become hoppers. They pass through five larval - or hopper - stages and become progressively more like the adults, which are readily distinguished only by the complete wings and genitalia. Growth continues for some time in the adult stage, especially in the exoskeleton and muscles, before sexual maturation begins. Once sexual maturity is reached, the adult females begin laying. The model distinguishes four age classes of adult females, in order to deal with the fact that the ratios of diapause to non-diapause eggs laid may change with progressing age.

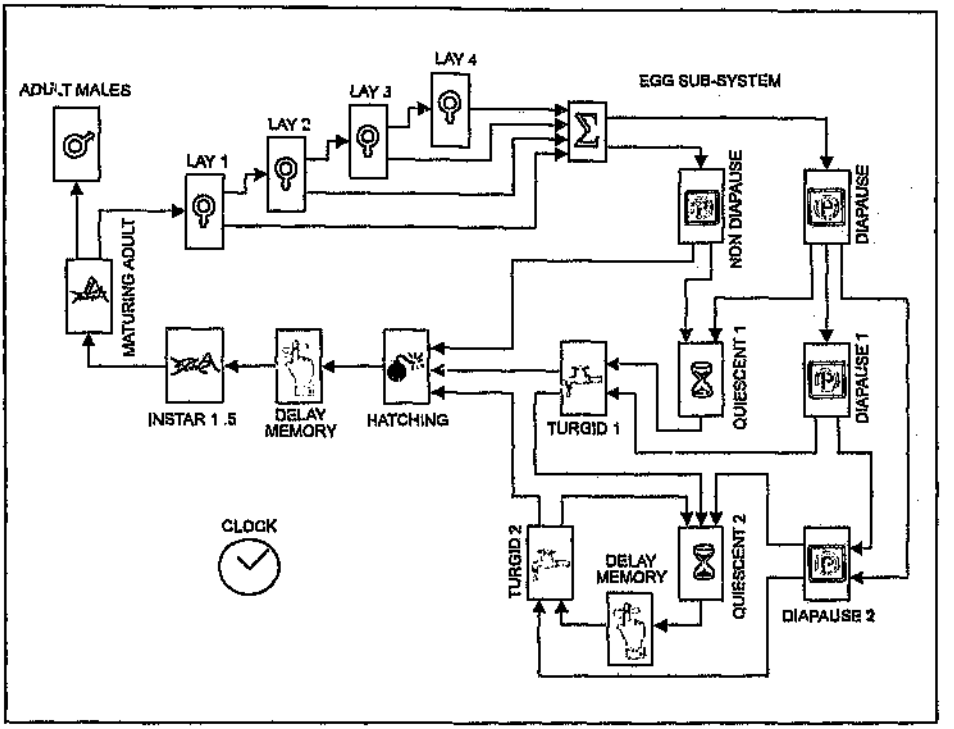


Fig. 3.3 The complete locust lifecycle block diagram. See also figure E5.1.

Modelled adult males are assumed to be available in sufficient numbers for mating, and their numbers are available for tallying in swarm totals. The sex ratio of mature adults is 1:1. Although mating occurs regularly thoughout the females' life, one insemination can suffice to fertilise all the eggs the female will lay in her lifetime (Norris, 1954, in Price 1988), which may total more than a thousand but is usually much less.

Development may be inhibited (slowed) in any stage of the life history, for days, weeks or months (in the case of some egg stages), during unfavourable seasons. The egg stage usually continues through cold winters, and eggs may remain dormant for several years. In some species, hibernation occurs in the hopper or adult stage, during mild winters, but in the case of the Brown Locust the hopper and adult populations usually do not survive the cold, dry winters. The lifecycle model incorporates a routine to eliminate adults and hoppers during the cold winter period.

HOW THE LIFECYCLE SIMULATION OPERATES I : LOOP STABILITY

The lifecycle simulation operates in discrete time increments of one day. Inspection of the block diagram (Figure 3.3) shows that it consists of a closed algebraic loop. Problems arise when trying to evaluate conditions in such a loop, since each stage requires an unambiguously defined input in order to evaluate the

output. Each stage derives its input from the preceding stage's output. If one progresses backwards around the loop in this way, a point of paradox is reached. At some point, information which we wished to compute is found to be dependant on that very information, and the cause/effect argument collapses. Under such conditions, Matlab will attempt to perform a Newton-Rhapson iteration, in order to find the points of stability within such a system. Under conditions where the loop can be evaluated (stable, convergent loop), the iterative process dramatically slows down the simulation. Matlab will allow only 200 iterations before declaring the loop unstable or insoluble. If the points of stability cannot be evaluated within 200 iterations, an error is signalled, and the simulation is halted. This method invariably caused "random" errors in the simulation. The precise nature of the errors has not been adequately explained. It is suspected that such loops have multiple points of stability, and fluctuations in conditions cause the iteration to converge to zero - an obvious, but not unique point of stability. Different initial conditions result in alternate points of stability being found - hence the apparent randomness to the error. A solution to preclude these loops has been incorporated into the simulation.

By inserting a single-step (= one day) delay into any recursive loop, an initial point of stability is automatically defined, since there is no dependence of any output on its own input at the same time. This is the recommended method of dealing with closed loops, according to the Mathworks (Programmer's Reference Guide, 1992). Two such delay blocks can be seen in the complete lifecycle block diagram (Figure 3.3).

The lifecycle is described by eight distinct developmental stages: eggs, hoppers (instars), maturing adults, four stages of sexually mature females, and sexually mature adult males. Instars one through five are not distinguished individually, since there are no events or processes exclusive to any particular instar stage. Adult females progress through four laying stages; each is represented. Distinction between males and females is made at the end of the maturing adult stage. Although biologically this is untrue, it facilitates a more compact model design, with no error introduced. Hopper stages require time to mature. This time is dependent on the amount of heat energy they receive. Based on the air temperature - measured on a daily basis - the degree day contribution to the hoppers is computed (Zalom *et al*, 1983) and used to determine how long they remain in this class. Adults mature on a time basis, without regard to heat input.

Each developmental stage is a self-contained object, with a well defined interface. As long as the interface structure is maintained, any changes or additions may be made to the object without affecting any other part of the model.

The egg stage is further subdivided into eight developmental stages. Eggs are laid as either diapause or non-diapause. Non-diapause eggs may hatch after sufficient development, if conditions are suitable, or proceed to a state of quiescence. Diapause eggs never hatch directly, and may progress to states of deeper diapause or quiescence. Diapause and quiescent eggs may need to undergo a process of rehydration before they develop further, providing there is sufficient warmth for continued development. Such rehydrated, or turgid, eggs reach catatrepsis and will die if a dry spell ensues. All egg stages require time to

develop to subsequent stages. This time is dependent on the amount of heat energy they receive from the soil. Based on the soil temperature - measured on a daily basis - the degree day contribution to the eggs is computed and used to determine how long eggs stay in each class. The factors which break diapause are not yet well understood. An unusual feature of the diapause eggs is that a small fraction enter a state of deeper diapause.

The egg subsystem is not visible to the top-level of the lifecycle model hierarchy. Eggs - tagged as either diapause or non-diapause - are passed into the egg sublevel, and all processes regarding egg development are invisible. Hatchlings are returned to the top-level of the lifecycle from the egg subsystem, from whence they develop as hoppers.

HOW THE LIFECYCLE SIMULATION OPERATES II: DATA PROCESSING

The simulation model has been optimised for speed at each possible stage. To achieve this, a simulation is divided into two steps. The first incorporates all the processes required for data preparation that do not constitute the simulation proper. These are processes that are required only once, no matter how many times the simulation proper is to be run. The second step is the simulation proper, which relies on the pre-processed data generated in step one. It is envisaged that users will want to perform multiple simulations consecutively, possibly modifying lifecycle- or simulation parameters. The two-step approach makes this fast, since manipulations that are required only once are performed in step one.

Individuals are categorised in the lifecycle according to the stage of development they have reached. Within each such stage, an age record of each individual is kept. Each lifecycle stage functions more or less as a first-in-first-out (FIFO) queue. Exceptions to this FIFO rule are when certain individuals in a lifecycle stage die (e.g., dehydration of catatrepsis eggs). The order of individuals within the queue cannot be reshuffled, but portions of the queue can be removed (proceed to next development stage) or deleted (death).

An example of such a queue is shown below, figure 3.4.

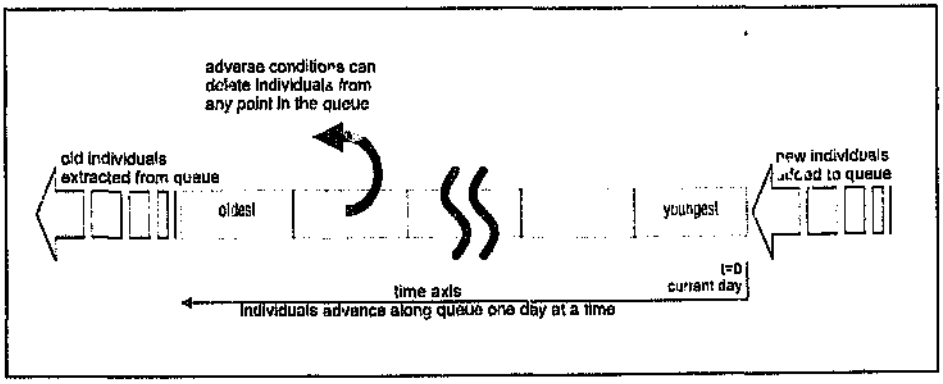


Fig. 3.4 First-in, first-out (FIFO) queue schematic.

The time axis shown in the figure runs from right to left, which is unusual. Matrices are conventionally augmented to the right, meaning that new individuals are added on the right-hand-side of an existing matrix. Thus the oldest individuals will be found in the left-most columns of any matrix, making the direction of increasing age right-to-left.

As the simulation progresses, the number of individuals in each stage of development is dumped to a record of populations. This record is updated on a daily basis. Once the simulation is complete, or terminated, there exists an entire history of the number of individuals present in each lifecycle stage, on each day of the simulation. This record may be plotted graphically, or used to perform statistical tests on the simulation accuracy.

The queue representing the lifecycle population for that particular stage, is a dynamic variable. It has no predefined size, and is redimensioned appropriately as the population size fluctuates. The population record is maintained in such a flexible data structure, rather than a set of simple variables, in order to minimise resource usage. In fact this was a prominent failing of the original Pascal model (Nalland & Hanrahan 1993), where the data structures were declared a fixed size. That requires allocating memory for the maximum anticipated population scenario. Problems arise when these anticipations are exceeded.

At any time, individuals within a queue may be deleted if conditions become unfavourable. Small portions within the queue may be nulled if the cause of death is an age-specific factor (dehydration of catatrepsis eggs, for instance). Such deaths are represented by the corresponding spaces being filled with zeros. Deleting the values and collapsing the queue is not an option, since it implies the sudden maturation of individuals younger than those deceased. All queues are continuously monitored to truncate them to their shortest possible length. Trailing zeros, although harmless as far as the simulation accuracy is concerned, are wasteful of memory.

The advantage of using a two dimensional array, in place of a set of scalars, is that the length of the array can be used to represent age, again reducing the number of individual variables that need to be created and updated. Once any initial resistance to understanding such two dimensional arrays it overcome, they become a very elegant and concise method of visualising the populations.

STATE VARIABLES

The data structures that represent various lifecycle stage populations are known as the *state variables* of the system. The state variables are those variables that are described quantitatively and are modelled. They are distinguished from the *environmental variables* which drive the system, and various internal variables that allow the model to execute but do not necessarily represent a real or tangible phenomenon. The state variables uniquely describe the system at any time, but interestingly, the state variables themselves are not unique; i.e., there are an infinite number of sets of variables that could be used to uniquely describe the system, not all of them representing real or physically meaningful quantities. Although state and state-space analysis has come into vogue recently among

modellers, the methods date back to the 1890s when Jules Henri Poincarè used state-space representations in his work on modelling celestial mechanics

Formally, the state of a system is a (non-unique) mathematical structure, containing *n* variables,
$x_1(t)$, $x_2(t)$, $x_3(t)$, ... $x_n(t)$, called the state variables, which together with the initial values $x_i(t_0)$ of this set, and the system inputs, is sufficient to define uniquely the system's behaviour for all future time.

As confusing as this statement may sound to the non-mathematically inclined, it has profound implications as far as building and running the simulation model is concerned. In short, the state is the minimum set of variables (state variables) which contain sufficient information about the present and past history of the system to permit computation of all future states of the system, assuming all future inputs are known.

Thus for the simulation model to be workable, three things are essential:

- the state variables must be defined.
- the future inputs (weather) must be known.
- the initial conditions must be specified.

To return briefly to the attractiveness of linear systems, as compared to non-linear systems: the general n^{th}-order differential equation that describes *any* system:

$$a_n \frac{d^n y}{dt^n} + a_{n-1} \frac{d^{n-1} y}{dt^{n-1}} + a_{n-2} \frac{d^{n-2} y}{dt^{n-2}} + \cdots + a_1 \frac{dy}{dt} + a_0 = u(t)$$

where y(t) is the system output, and u(t) is the system input,

can in the case of a *linear* system, be decomposed into *n* (non-unique) first order differential equations of the form:

$$\frac{dx_n}{dt} = F\{x_n(t); u(t); t\}$$

where F is some vector function of x_n, u, the input, and time, t. x_n are the *n* state variables of the system. Since these can, by definition, in conjunction with the system input, describe the system for all future time, we can easily determine the system output by:

$$y(t) = G\{X_n(t); u(t); t\}$$

where G is some vector function of the state variables x, the system input u, and time, t.

CHAOTIC BEHAVIOUR DESCRIBED

Chaos can be loosely described as aperiodic (random looking) dynamics from a [stable] deterministic system sensitive to initial conditions, such that close initial points will diverge from each other at an exponential rate. Since the initial conditions cannot be specified with an absolute degree of accuracy, no chaotic

system can be predicted for long without updates from the real system - even if the model is perfect! (Allen, 1992)

To further confuse the issue, non-linear systems can exhibit *quasi-periodic* behaviour, which superficially appears to be chaotic, but is not. Quasi-periodic systems are characterised by the oscillatory modes having frequencies that are irrational multiples of one another, i.e., they are anharmonically related (Moon, 1987). The distinction of chaotic systems lies in the fact that the curve of intersection of their response with a line (2 dimentional system) or plane (three dimensional system) is characteristically fractal. This curve of intersection is known as the Poincarè map, but details of its characteristics are beyond the scope of this report (see Moon, 1987).

While complicated dynamics are possible in very simple systems, certain characteristics of a system do favour chaos. Delayed non-linear feedback is a prime example, as is periodic forcing of a non-linear system. Inspection of the locust lifecycle block diagram, and consideration of the model operation will show without a doubt that these two chaos-conducive criteria are present. Allen (1992) states that virtually all biological systems fit this description. It is thus not surprising that the locust simulation model exhibits chaotic behaviour, although formal proof of this requires more rigourous analysis than has been attempted by the current author.

APPLYING THE RULEBASE

The simulation operates in discrete time increments of one day (24 hours). This coincides conveniently with the interval of rainfall records, and temperature records. Conceptually, all changes that occur within a time span less that 24 hours are assumed to occur within a single instant, once per day. It is conceivable that this might introduce some errors, since actual events proceed on a continuous time scale of infinitesimal resolution. Increasing the simulation resolution amounts to more than simply adding extra records to the weather data (twice- or thrice daily measurements) since the rulebase implicitly assumes a timestep of 24 hours. Specifically, computations based on triangular modelling of daily temperature would require reworking, as would the degree-day contribution calculations.

The rulebase for any particular lifecycle stage performs three activities:

1 It evaluates the current states of development reached by individuals, or groups of individuals within that stage, and adjusts those states of development as necessary. This includes actions such as augmenting the accumulated degree-day total, increasing the age of the individuals universally, setting a catatrepsis flag if appropriate, and so on.

2 It deletes individuals that have succumbed to unfavourable conditions, such as waterlogged eggs, individuals that are too old, or catatrepsis eggs that have dehydrated.

3 It promotes individuals to subsequent stages if and only if they meet the necessary requirements for further development, such as sufficient accumulated degree days, or eggs rehydrating, etc. In the case of adult females, each stage is allowed to generate eggs.

These rules are all based on if/then constructs, and the decision points are determined largely on the numeric values of the lifecycle parameters selected by the user. Rules used in the model are outlined explicitly in Appendix F. The reader is urged to interpret these in conjunction with the model block diagram (figure 3.3).

INITIAL CONDITIONS

The source of individuals in any particular stage is one of two. Individuals may be introduced into any stage from the preceding stage. In the case of some egg classes (viz. quiescent 1, quiescent 2, diapause 2, turgid 1, turgid 2) there exist multiple preceding stages, hence individuals may be introduced from one or two preceding stages. Individuals are always moved on the current day. Individuals are never moved into the history portion of any stage, meaning that they are not considered to grow older during a migration event. Migration is a relatively brief event in the context of the insects' life and therefore regarding it to occur instantaneously introduces negligible error. The alternative source of individuals to a lifecycle stage is from the record of initial conditions. Logically, this is only applicable to day one of the simulation, but the initial condition record has the facility to store a history. The initial conditions are a record of all the populations of all the lifecycle stages on the zeroth day of the simulation. Since it is not possible to run a simulation from the beginning of time, it is convenient to establish that on day zero (which can be any convenient starting reference date) there exists a set of populations defined by [x]. [x] is a user editable matrix. Although [x] stores information for day zero only, it can, as noted earlier, reflect a history. In other words, on day zero there are individuals that arrived in that lifecycle stage on day zero - and are therefore zero days old, but there may also be individuals that are one, or two, or three, or more days old. As the simulation proceeds, these older individuals will advance to subsequent stages before younger individuals.

In this way, a simulation can be started at any time, and will emulate a much longer simulation - stretching further back in time, provided that the simulation state at the discontinuity is known.

Recall that chaotic systems are exquisitely sensitive to small changes in their initial conditions (Moon, 1987), so for accurate simulations it is very important to use realistic initial conditions. Techniques for estimating suitable values are described in the following chapter on model testing.

Initial conditions are only read at the start of the first simulation of a series, or upon request. Subsequent simulations will build on populations synthesised in the previous simulation run. Under certain circumstances, the user may wish to run several simulations, using the same initial conditions for each run. If lifecycle- or migration parameters are to be changed and their effect investigated, it is necessary to use identical starting populations for all simulation runs, so that

results may be directly compared. In such instances, initial conditions can be restored upon request.

The reason that initial conditions are not read at the start of every simulation is so that long simulations can be performed in several parts, without distortion of population numbers at the hiatus points.

Furthermore, it is possible to create a library of initial conditions. Thus several different sets of initial conditions may be named and stored independently, for various simulations runs requiring a diversity of starting populations.

3.3 INCORPORATING MIGRATORY PHENOMENA

Migration involves the translocation of certain portions of the population at a specific geographic location to another distinct geographic location. Migration is regarded as a short duration event - in terms of the timescale of the lifecycle - hence no development is considered to occur during a migration. Individuals leaving any site will arrive at other sites at the same stage of development they had reached when they left.

Only individuals capable of flight can migrate, so the only development stages communicating with the migration engine are the sexually mature adult females. Adult males could possibly be incorporated into this scheme, as could the maturing adults. Hoppers exhibit a phenomenon of marching, but since they cover relatively small distances compared to fledglings, with no obvious environmental influence over the direction in which they move, no account of this is taken by the simulation model. Maturing adults are known to migrate (Rainey, 1989), but they will not be responsible for eggs being introduced to the destination site (this is already taken care of by migration of adult females). The model does not currently allow migration of maturing adults in order to keep the migration engine as simple as possible. The primary outcome of allowing maturing adults to migrate would be changes in the timing of immediate population interchanges, but little change to to the pattern of egg redistribution. Currently, the maturing adults require around 60 degree-days before they can progress to the migrating adult stages. Considering that typical Karoo summer temperatures are in the upper 20 - 30°C range, and the lower threshold for development is approximately 20°C, 60 degree-days is accumulated in approximately one week. Maturing adults would then conceivably require at least some development before migrating, and thus advance migration events by maybe 3 - 6 days. In practice, it is difficult to calibrate the model to this degree of temporal accuracy because actual outbreak records are monthly cumulative totals. Once the timing of outbreaks is being modelled satisfactorily, it will be appropriate to look at the maturing adults as a component of swarms.

Recall that each lifecycle stage queues individuals as they enter that lifecycle stage. During migration, the numerical contents of the adult female queues are modified twofold:

Firstly, a proportion of the contents of each queue is deleted. This represents emigrating individuals that leave for another station or stations.

Secondly, a calculated quantity of individuals is added to the appropriate queue. This represents immigrants, that have arrived from another station or stations.

Individuals that migrate are introduced into the lifecycle stage at the destination site as newcomers to that stage. They are regarded as having the same history as those individuals that have been introduced into that stage from the previous stage at the same site. Migrations do not facilitate the transition from a developmental stage to a subsequent developmental stage at the destination site.

This is done because the lifecycle model considers that females lay eggs upon being introduced to a subsequent development stage. I.e., a female progresses, after *n* days, to the subsequent lifecycle stage. As soon as she arrives in that next stage, she lays. While this is not strictly true biologically, it makes implementation of the model easier. The omission of the usual growth period prior to laying may introduce a negligeably small error in the timing of egg laying. This error can be substantially compensated for by tuning the accumulated degree-days required for progressing to the next stage. She will then remain in that stage, not laying again, until sufficient degree-days are accumulated for her to proceed to the next stage, where she will again lay. Thus, emigrating females are considered to have laid at their departure site, and will also lay at the destination site. A time - denoted minimum_age_to_fly - is required to elapse before a female may leave any lifecycle stage by migration. Thus she cannot lay on consecutive days at different sites (unless minimum_age_to_fly has been set to zero or one).

Only gregarious females are considered to migrate. Solitary individuals do not leave the site at which they hatched (Backlund, 1955). The ratio of solitary to gregarious females is determined at the time of adult maturation. This ratio is user-adjustable, but does not change automatically as population densities fluctuate. Wolda and Dennis (1993) suggest that density dependent computations only become significant after 20 - 30 generations, and in the case of the Brown locust there is little field data to suggest under what conditions gregarisation is initiated.

The minimum flight age is not the only criterion that determines whether a female may leave a site. The wind conditions on that day are of paramount importance.

THE EFFECT OF WIND ON THE MIGRATION PATH.

The reader is referred to Chapter 2 for background information on how wind conditions control and influence migratory events. The salient points are listed below:

- Migrations occur predominantly in a downwind direction.
- Migrations occur only if temperatures are sufficiently high.
- The flight speed is largely irrelevant.
- The insects' speed of movement is the same as that of the prevailing wind.
- The wind data records are only applicable to the instant at which the measurement was taken.

The migration engine developed incorporates features to take account of all these phenomena.

Establishing whether conditions conducive to migration are present is *partly* a case of looking at the air temperature, and the number of gregarious females present at each particular laying stage. What this really indicates is that the locusts are likely to start to fly, it doesn't necessarily mean there will be large scale migrations. To establish *that* requires a look at the prevailing winds.

Two factors are important when considering the influence of the wind.

Firstly, the direction is obviously important since it is this that will have the greatest control over the trajectory followed by the locusts. The destination site is determined by the direction of the prevailing winds at both sites. Secondly, the wind speed is a key factor in determining the distance over which the migration can occur, since time of flight is limited.

The locusts' actual flight-speed is largely irrelevant, since they are, for the most part, at the decree of the wind. In general, the distance travelled, denoted s, will be some function of the product of the wind speed, denoted v, and the flight time, denoted t.

by definition, velocity is the time derivative of displacement
that is $v = ds/dt$
hence $s = \int v.dt$

The velocity is assumed to be a constant during the simulation interval increment. This is completely valid, since there is only one wind speed measurement applicable per day. Wind speed data is available at 08h00, 14h00 and 20h00, but since the simulation time increment is 24h, the different values cannot be processed during the simulation. At best a compromise can be reached by fitting a curve - polynomial or otherwise - through the three measurements Then an appropriate value could be read off the fitted curve, correspondence to the preferred time of flight. This process would have to be performed on the weather data prior to commencing a simulation run. Alternatively, a single measurement can be used, that being the one that most closely matches the observed flight time, viz, shortly after sunrise. The latter approach has been adopted in the current model.

Thus $s = v\,t$

where v is the early morning wind speed.

The time of flight, t, is essentially determined by temperature and possibly by changes in light intensity. Light intensity is likely to be sufficiently high a considerable time before the daily temperature starts rise. For this reason temperature is considered the limiting factor and has been used exclusively. Using a triangular-wave approximation to the sinusoidal daily air temperature fluctuation, that portion of the day where the temperature is above the threshold-temperature-for-flight is evaluated. This is shown in Figure 3.5 overleaf.

The day fraction is used as the flight time.

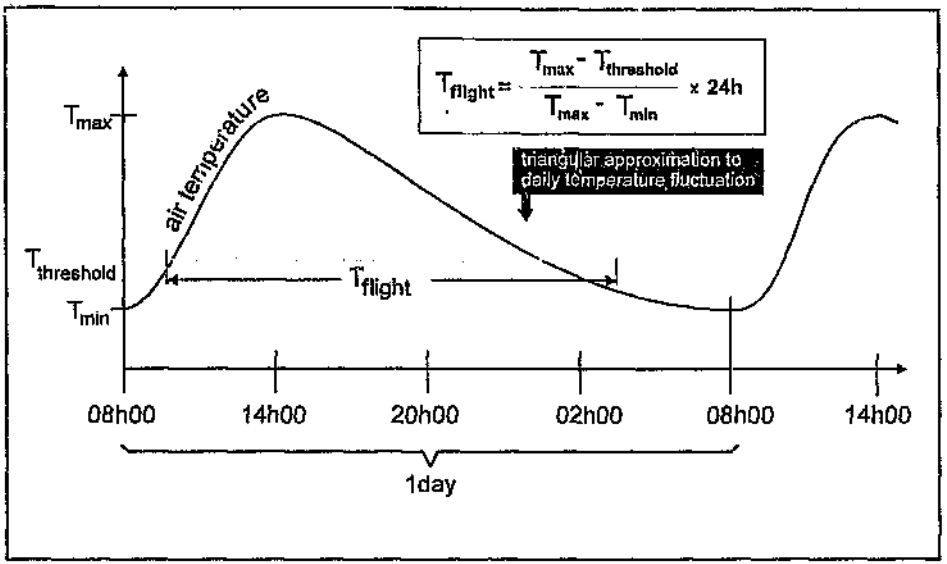


Fig. 3.5 *Time of flight estimation schematic.*

It is not an ideal measure, since if locusts have been lifted into high altitude weather systems they are often passively transported, even though the requirement of exceeding threshold-temperature-for-flight is no longer maintained. It must be stressed that we are dealing only with low altitude flight here. Once insects are carried up into the higher strata of the atmosphere, their behaviour has little dependence on the surface temperatures and wind data that we have available for consideration. However, consultants (Price, 1996 pers. comm., Napier-Backs, 1996, pers. comm.) have suggested that migrating swarms of Brown Locusts - in contrast with many other species - begin flying shortly after sunrise, and are confined to altitudes of around 10 to 20 metres above ground.

Inspection of Figure 3.5 will reveal that if the threshold temperature for flight lies below the daily minimum temperature, then the entire 24-hour day will be perceived as conducive to flight. It is not certain whether temperature alone governs cessation of flight, since it has been suggested that visual cues play a rôle in flight mechanisms. Nonetheless, nightime migrations are known to occur in some instances.

WIND DATA PROCESSING - PROBABILITIES BASED ON WINDSPEED

Direct knowledge of the expected translocation distance, s, is not particularly useful though, since we are dealing with a small number of fixed weather stations, and there may well be no other stations at a radius equal to s. In other words, one can determine that the locusts will fly a distance s, but what does one do if that distance doesn't correspond to another site being modelled? What is more pertinent is how closely that flight distance matches the actual distance between the two stations of interest. If the wind speed is low, or flight time is short, then

there is a likelihood that the locusts will not be carried far enough to reach the downwind station.

Conversely, if the wind speed is high, or flight time long, then there is a possibility that the airborne locusts will be carried right over the downwind destination sites, and few will be deposited. Hence, in terms of the model operating at two fixed points, there is an optimum flight distance (viz. it matches the interstation distance) that may or may not be achieved, depending on conditions. This phenomenon is illustrated in Figure 3.6.

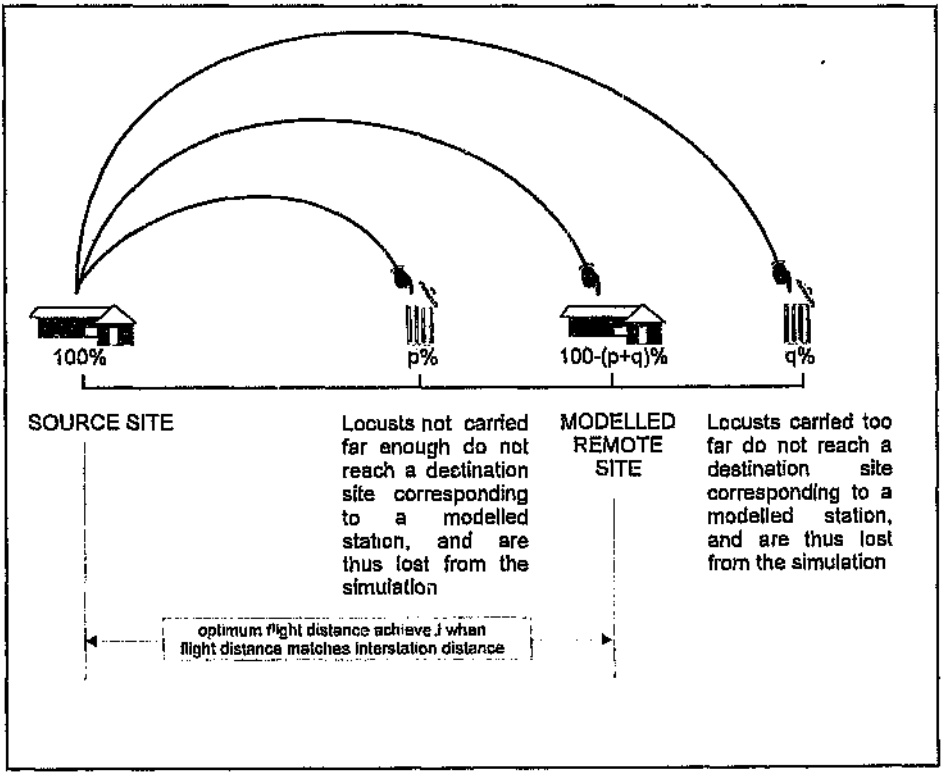


Fig. 3.6 Source & destination sites; loss of individuals to unmodelled sites.

Such a function may be conveniently modelled by a curve of the form

$$P = \alpha . e^{-\beta (s-s')^2}$$

which is the Gaussian probability distribution (Davies & Hicks, 1975). α and β are constants incorporating the standard deviation and variance of the random variable, s. A suitable form of the function is shown in Figure 3.7 overleaf.

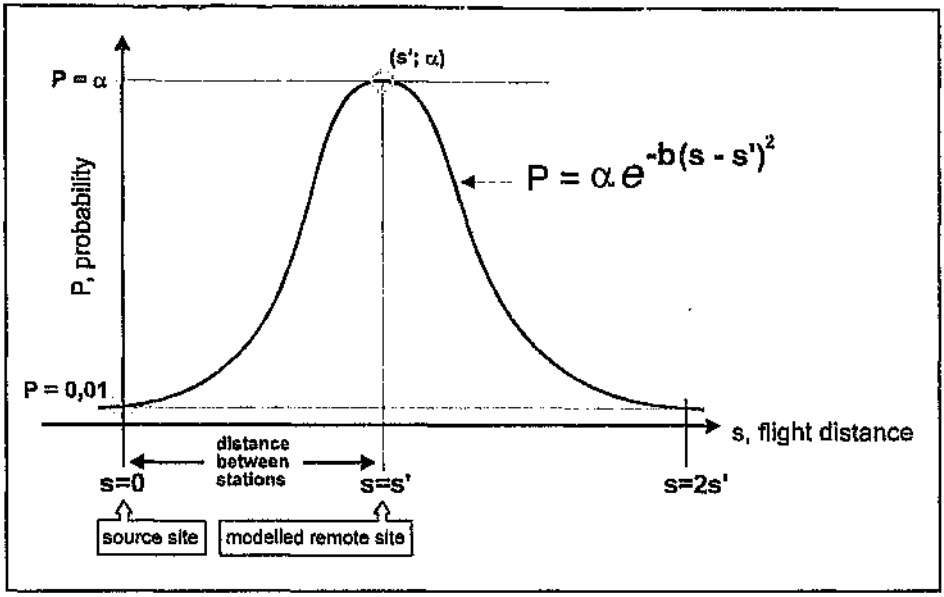


Fig. 3.7 Interstation distance and associated probabilty.

s denotes the actual flight distance, the product of wind speed and flight time. The computation of s has been described earlier.
s' denotes the distance between the two stations of interest.

The vertical axis, P, represents the probability of locusts being deposited at a certain defined distance s . The probability, P, is a function of s, the translocation distance, so P is more correctly represented by P(s). For every P(s) we consider s' to be a constant. P(s) - in the form finally used - is not strictly a *de facto* PDF, this being described more fully overleaf.

The turning point of the curve passes through the point (s',α) where the distance travelled during the flying time is exactly equal to the distance between the stations , s' . That is

$P(s)|_{s=s'} = P(s') = \alpha$

From this equation it can be seen that when the flight distance, s, exactly matches the interstation distance, s', then there is the highest (=α) probability that the locusts will be deposited at the downwind station.

If there is no wind, no migration is expected, and so P(s=0) might be regarded to be zero. However, a probability distribution curve - being an exponential function - never attains a P value of zero. It is possible to shift and scale the curve to force it to pass through the origin, but in an attempt to accommodate the limitations of the wind speed records, the curve has been forced to pass through the point P equal to one percent, where s is zero. That is

$P(0) = 0{,}01$

Forcing the curve through the origin by P-axis shifting is likely to cause conceptual difficulties since P(s>2s') and P(s<0) will be negative quantities.

In summary, if the flight distance, s, differs largely from the interstation distance, s', then there is a very low (= 0,01) probability that locusts will be deposited at the downwind station. A lower threshold probability of 1% is used in place of zero, to allow for migrations even when wind velocities are not in accordance with the required distances. This has been done because the wind speed records are of limited resolution (very low wind speeds are recorded as zero, because the records are rounded to the nearest multiple of 5 knots). Also, wind speeds may fluctuate considerably during the course of the day, so no wind measured at 8am does not imply that there will not be a light breeze later during the morning.

It is convenient to let α=1. Strictly, assigning a value of "1" to the turning point is mathematically incorrect, because the definition of any true probability function requires that the *total area* under the PDF curve be unity. Since the curve extends from $-\infty$ to $+\infty$ along the random-variable axis, it is impossible for the area to be unity unless the turning point of the curve is somewhat less than unity. However, in the case of estimating probabilities for locust flight distance (and later, flight direction), what is important is the *shape* of the distribution curve. Hence the use of an expontential square (which it shares with the Gaussian, or normal distribution). By avoiding having to maintain the correct area under the curve, computations of the constant *b* (described below) become extremely simple. The error so introduced is compensated for (see later section "Combining the probability functions") by a multiplier *k*, which may incorporate the reciprocal of α.

If a curve encompassing unity area is required, then there are an infinite number of possible curves, all of which will satisfy the other requirement of passing through (0;0,01). The optimum curve would be selected on the basis of satisfying a specific variance. Note that then α becomes a function of s'. In practice, the variance associated with the flight distance and direction distributions is unknown, hence α and b are chosen for simplicity of computation, while still retaining the correct shape of the curve.

Reducing the P(0) value will have the effect of squeezing the curve on either side of s'. The actual curve used in the computations is:

$$P = \alpha . e^{-b(s-s')^2}$$

$$\alpha = 1, \quad b = \frac{\ln_e 100}{s'^2}$$

Manipulation of α and b allows the turning point of the curve, P(s=s') and the zero-wind probability, P(s=0) to be adjusted to accommodate any desired standard deviation.

CALCULATING THE P VALUES FOR WIND SPEEDS

The P(s) value is calculated for each day of the simulation.

For each day, there exists a record of wind velocity, v. Also, the air temperature maxima and minima are known, and by triangular approximation, the flight time, t,

can be found. P(s) changes on a daily basis, depending on the temperature, the temperature range, and the wind speed. The interstation distance, s', is obviously fixed, so the P(s) values are applicable to a particular pair of weather stations only.

The simulation incorporates a look-up database of all possible sites' latitude and longitude co-ordinates. For all possible simulation sites available, the simulation "knows" the geographic location of those sites, since this information constitutes part of the data required when adding a site name to the program menu. Thus s', the interstation distance is automatically computed from these co-ordinates when the two simulation sites are specified for any particular simulation.

IMMIGRANTS AND EMIGRANTS; THE SOURCE AND DESTINATION SITES.

Emigrants are determined as a function of the prevailing wind speed, and the temperature. These are the only two *climatic* factors that influence the simulation's decision whether or not emigration will occur and how profusely. This requires determination of the wind speed only at the station under consideration. That is, emigrants can be carried out of a site population, even if the destination is unknown. P(s) computed above is not useful in calculating emigrants from a site. Determining emigrants is a difficult procedure conceptually, since there is little information available to indicate at what wind speeds locust will begin to fly. The wind speed is only measured to an accuracy of 5 knots. Zero readings do not allow any emigration to occur, and 5 knots is certainly sufficient for take-off. Hence in practice, there is no real need to determine the exact speed above which emmigration is possible. The assumption is made however, that the higher the wind speed, the greater will be the number of departing emigrants.

A linear mapping relating quantity of emigrants to wind speed has been chosen for simplicity. In order to prevent mathematical incongruencies, it is required that the relationship between wind speed and quantity of emigrants never demands more emigrants than there are individuals in the population. To maintain this condition, the entire wind speed record is scanned prior to a simulation commencement. The highest wind speed that will be encountered during the imminent simulation is detected, and that is used as a limit when computing the mapping between wind speed and number of emigrants. At the maximum possible wind speed, a specific fraction of the population is assumed to leave. This number may be arbitrarily changed by the user. Values between 75% and 100% might be typical. This implies that when the wind attains its maximum speed, then maximum depopulation will occur (i.e., 75% to 100% of gregarious individuals will be removed). At all other wind speeds, a correspondingly smaller percentage of the gregarious population will be removed, down to a wind velocity of zero, when no individuals will be removed. Clarke, Ashall, Waloff & Chinnick (1969) in Lambert (1972) recorded an estimated average of 35% of locusts leaving observation areas per day during work done in Queensland. Without doing any rigorous calculations, but by inspection of plots of the wind records, the most frequently occuring wind speed represents between 33% and 46% of the maximum recorded speed at any particular site. Thus it seems reasonable that if Clarke *et al.* observed on a typical day, then the prevailing wind would be roughly 39% (the

average of 33% and 46%) of the maximum wind ever experienced at that site. The figure is close enough to the 35% depopulation fraction to justify using maximum observed wind speed corresponding to 100% depopulation as a locus for fitting our linear mapping.

Kisimoto and Sogawa (1995) have discovered that planthoppers only take off if the wind speed is below approximately 3m.s.$^{-1}$. There might be similar characteristic in Brown Locusts, but mention of it has not been found. If it is so, then it would be appropriate to apply a distribution curve to the wind speed, as has been done to determine interstation flight distance. Such a probability distribution curve would account for some optimal take-off wind speed, and simulate reduced take-off numbers if wind speed was particularly low, as well as if wind speed was excessivley high. A limit of 40kmh^{-1} for African Migratory Locusts is suggested by Rainey (1989), which corresponds to 21,6 knots, but at all the sites available over the period 1985 - 1990, the highest recorded morning wind speed was 15 knots, so no upper limit of wind speed for take off has been designed into the model.

In order to determine the influx of individuals, it is necessary to determine whether there are likely to be any individuals carried in on the prevailing winds. This is dependant on two factors:

- are there any stations that are in the same direction as the prevailing wind's source?
- are there any other stations that have prevailing winds such that this station is downwind?

These two are subtly different, in that they are based on different wind conditions at different stations. *Immigration is only likely if there has been emigration at a site* that is upwind of the site under consideration.

In most cases the wind directions vary on an instantaneous basis. The recorded wind directions are at best daily averages, or values that happen to be applicable at the instant the measurement was taken. In addition to this, the resolution of the available wind bearing is only one-in-sixteen (22½°).

A practical and convenient way to accommodate these uncertainties is to synthesise one or more wind direction probability functions.

WIND DIRECTION PROBABILITY FUNCTIONS.

For any pair of weather stations, it is convenient to define two pairs of probability functions:

$P_A(A \rightarrow B)$ = probability (based on wind data at A) that B is downwind of A

$P_A(A \leftarrow B)$ = probability (based on wind data at A) that B is upwind of A

$P_B(B \rightarrow A)$ = probability (based on wind data at B) that A is downwind of B

$P_B(B \leftarrow A)$ = probability (based on wind data at B) that A is upwind of B

Note that the first and the last shown here are really telling us similar things - the chance of getting insects from A to B. They are different though, because they are

dependent on different site data. The same is true for the second and third - which both show the chances of getting insects from B to A

The first probability, P_A(A *to* B), is of no value to a simulation occurring at A, but it is of value to simulations occurring at B, since it determines - in part - if any individuals leaving A might be on a trajectory directed toward B

The second probability P_A(A *from* B) is useful to a simulation running at A, since it determines - in part - if A is likely to be on the receiving end of any wind-borne locusts from B.

Similarly, $P_B(B \rightarrow A)$ is useful for a simulation running at A, but not useful to one running at B, and $P_B(B \leftarrow A)$ is useful for a simulation running at B, but not useful to one running at A.

CALCULATION OF P FUNCTIONS FOR WIND HEADINGS

Figure 3.8, below, shows a hypothetical example of two sites A and B, and an arbitrary wind direction, denoted omega, ω.

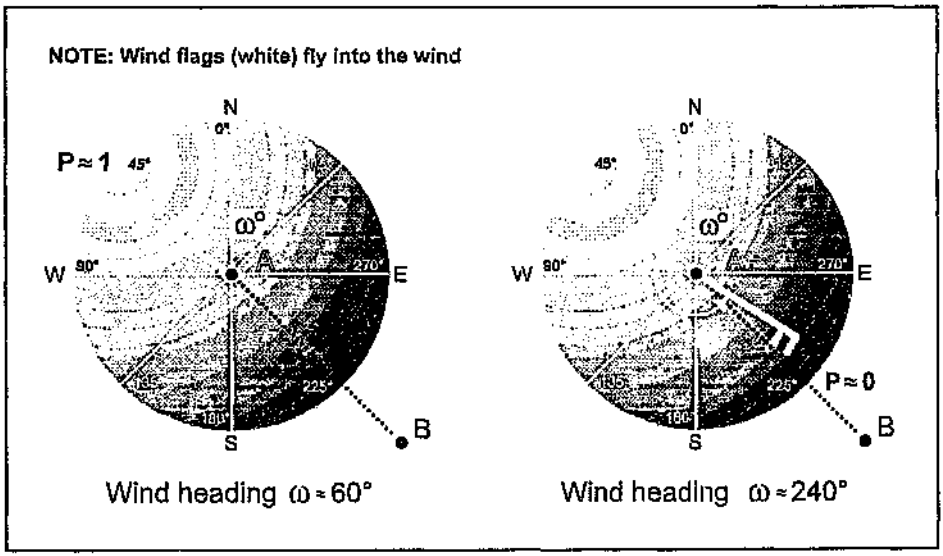


Fig. 3.8 Probability that station B will be downwind of A for any wind heading.

The wind heading is always measured relative to true north (zero degrees), and the heading denotes the direction from which the wind is blowing. B has arbitrarily been placed at a bearing of approximately 225° relative to A. (A is of course then at a bearing of 45° relative to B).

Consider the wind heading at A to be approximately 60° - as shown in the left-hand diagram of Figure 3.8. In this case there is a high probability that B is downwind of A, P≈1. This probability is denoted by the intensity of the shading of the disc. Bright colouring denotes high probability. If the wind heading changes, to say nearly 240° - as shown on the right-hand diagram, then the probability that B is

downwind of A is very low. The shading in this area of the disc is very dark to indicate this, $P \approx 0$.

Consider now the general case (Figure 3.9) where theta, θ is the bearing of B relative to A, and let the wind-heading - measured at A - be denoted omega, ω.

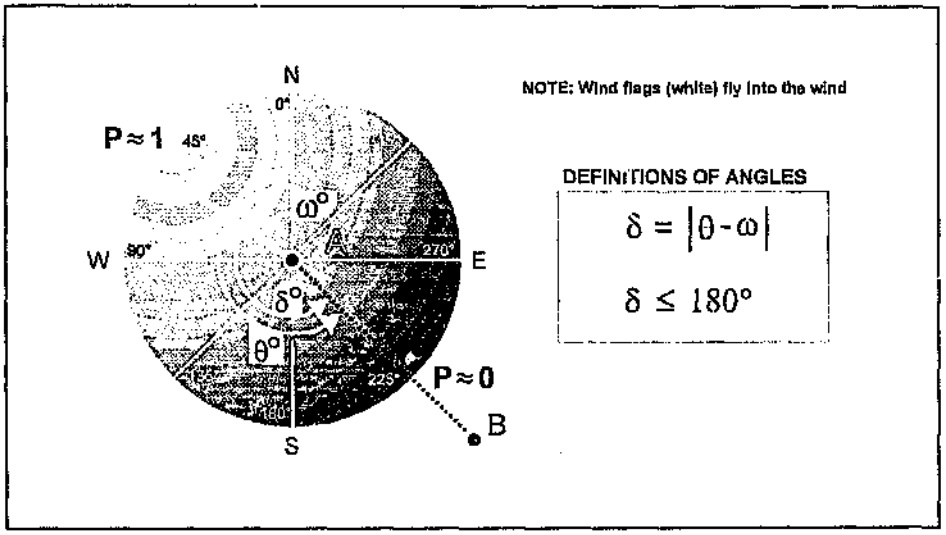


Fig. 3.9 Probability determined by the difference between omega and theta.

Suppose it is required to find the probability, or likelihood that B is downwind of A, for some wind direction ω. The probability function - suggested by the shading - is dependent on the difference between the bearing of B relative to A, θ, and the wind heading, ω. This difference may be called delta, δ. Intuitively, it can be seen that if δ is close to zero, then P will be close to zero. Conversely, if δ is close to 180°, then P will be close to unity.

Delta is constrained to be positive, and always lies between 0 and 180°. All that is of interest is how large the difference between ω and θ is. A negative value for δ can be interpreted in exactly the same way as the equivalent positive value, i.e., $\delta \equiv -\delta$. Thus P_A(A *to* B) and P_A(A *from* B) are both some function of δ, and δ is defined as the absolute difference between theta and omega:

$$\delta = |\theta - \omega|$$

Using the an exponential square probability distribution function once again, we can determine: P_A(A *to* B) and P_A(A *from* B) as functions of the angle δ. As before, the constant b scales the curve to pass through the desired P(0)=0,01 point.

$$P_A(A \rightarrow B) = \alpha\, e^{-b(\delta - 180°)^2}$$

yields a curve of the appropriate form, where $b = (\ln_e 100)/180^2$

Once again, it is convenient to let $\alpha=1$, and correct this error later by an inverse multiplying factor, included in an all-encompassing term *k*.

This probability curve is shown in Figure 3.10.

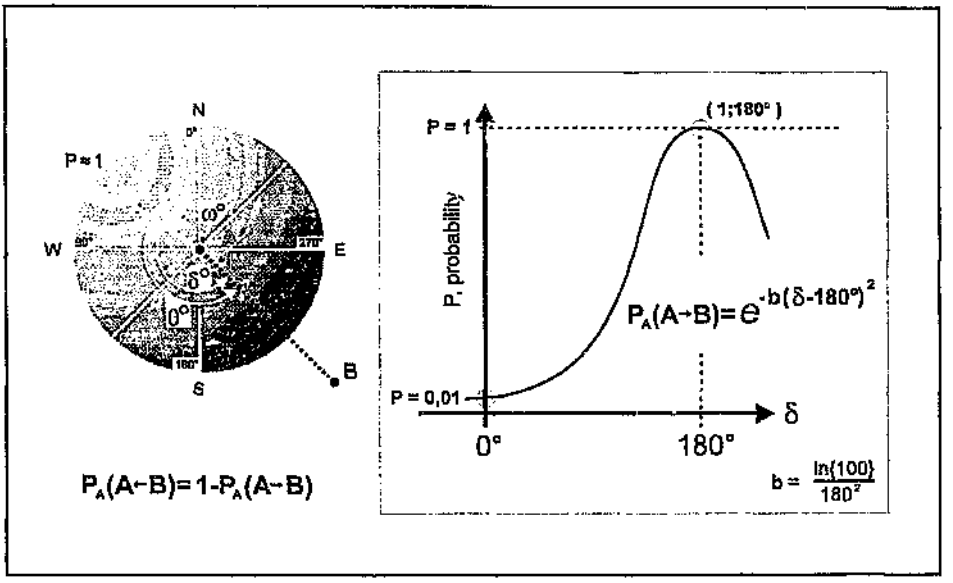


Fig. 3.10 Angular PDF: probability that station A is upwind of station B.

Note that $P_A(A \leftarrow B)$ is simple to calculate from $P_A(A \rightarrow B)$, since

$P_A(A \leftarrow B) = 1 - P_A(A \rightarrow B)$

Again, these probabilities vary on a daily basis, as the wind bearing changes.

$P_B(B \rightarrow A)$ is computed in exactly the same way, by evaluating delta for each day's wind direction measured at station B. $P_B(B \leftarrow A)$ can easily be found from $1 - P_B(B \rightarrow A)$

Each of the four P(A, B, δ) functions is calculated for each day of the simulation and for wind measurements at each of the two stations.

THE VON MISES CIRCULAR PROBABILITY DISTRIBUTION

The approach described above, using a distribution based on the exponential square of the angle, can be more accurately performed using a von Mises circular distribution, which is the most common model for unimodal circular data (Fisher, 1993). This distribution is the usually encountered analogue to the linear normal distribution (Batschelet, 1981), and takes the form of an exponential cosine. In the case of 180° (π rads) being the mode, or centre, of the distribution, the PDF takes the form:

$$P_A(A \rightarrow B) = \alpha\, e^{\kappa \cos(\omega - \pi)}$$

$$\alpha = \frac{1}{2\pi\, I_0(\kappa)} \qquad \text{and} \qquad I_0(\kappa) = \frac{1}{2\pi} \int_0^{2\pi} e^{\kappa \cos(\omega - \pi)}\, d\omega$$

Note that the term $I_0(\kappa)$ serves to scale the magnitude of the probability function and ensure that unity area is encompassed.

κ is the parameter of concentration, and determines the width of the bell-shaped curve about the mode.

To achieve the same conditions as before, $P(\pi) = 1$, and $P(0) = 0.01$, the required values are $\alpha = 0.1$ and $\kappa = \ln_e(10)$. Note that strictly $P(\pi) < 1$ for unity area, hence the need once again for a later correction term in *k*.

Although the von Mises equation differs in form from the *ad hoc* approach described earlier, the two equations yield a substantially similar curve when plotted as functions of ω for the range 0 to 2π (=360°). This is shown below, where the two forms are superimposed on the same axes. Note that here the von Mises equation has a substantially larger κ value than $\ln_e(10)$, which results in the probability dropping below 1% at the extrema of ω. Generally the von Mises distribution is wider that the "exponential squares" form of distribution.

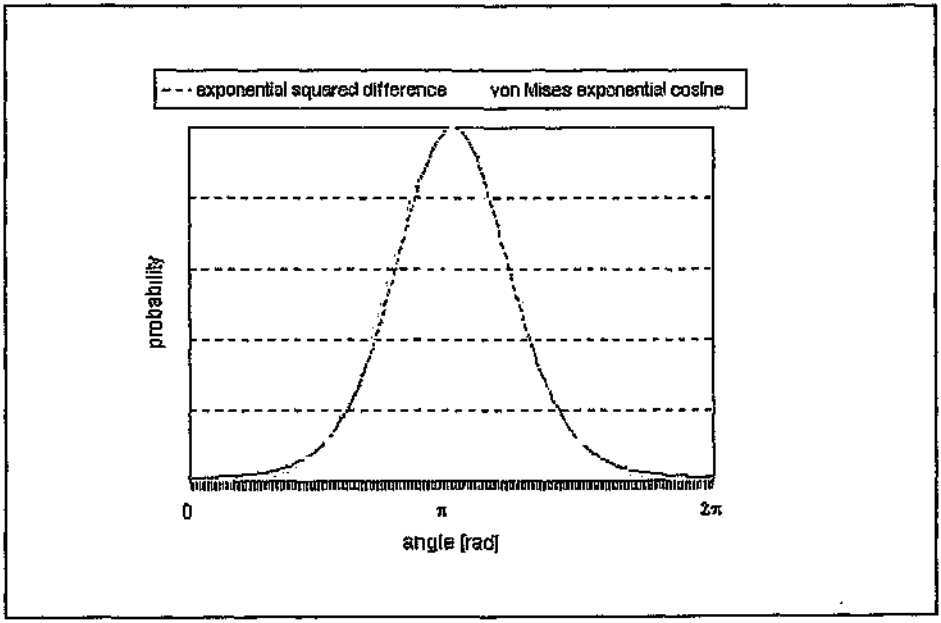


Fig. 3.11 The von Mises probability distribution compared with the distribution obtained by using the term $e^{-b(s-s')^2}$.
Note the von Mises distribution (as $\omega \rightarrow 0; 2\pi$) tends to zero faster under conditions of equal width, as shown; alternatively, for equal P(0) values for both distributions, the von Mises curve is substantially wider (not shown).

COMBINING THE PROBABILITY FUNCTIONS

Keeping in mind that in the first instance we are working only with a two-node model, for *each* weather station node we have

- The probability that this station is downwind of the other station - based on this station's wind direction: $P_A(A \leftarrow B)$ if this station is A, and the other is B.
- The probability that the station is downwind of the other station - based on the other station's wind direction: $P_B(B \rightarrow A)$ if this station is A, and the other is B
- The probability that locusts will be deposited at this station, based on the current day's wind speed, the expected flying time, and the distance from the other station: P(s). This probability is computed for wind speeds at each of the two stations and is thus more correctly denoted $P_A(s)$ and/or $P_B(s)$, and hence two additional probabilities arise.

Thus **four** probabilities are computed for any potential migration between A and B.These probabilities are combined into two probability indices, one for each station, to determine immigration numbers to that station.

For a lifecycle simulation running at station A:

Insects arriving at A: $P_B(s) \times P_B(B \rightarrow A) \times P_A(s) \times P_A(A \leftarrow B) \times k$

For a lifecycle simulation running at station B:

Insects arriving at B: $P_A(s) \times P_A(A \rightarrow B) \times P_B(s) \times P_B(B \leftarrow A) \times k$

In each case, these probabilities are used as scaling factors on the number of gregarious females, beyond the minimum flying age, added to each adult female lifecycle stage. The number of emigrants from a distant station is multiplied by the product of the appropriate scaling factors, and is then added to the appropriate queue for the local station. In this way there can be emigration from the distant station, but that does not imply immigration at the local station, since if the winds are "wrong" (probabilities are close to zero), very few insects leaving a source site will arrive at the other modelled site. Conversely, there can be no immigration at the local station if there has been no emigration from the distant station (in the case of a two site model). The operation of the migration engine is represented schematically in figure 3.12, overleaf.

k is a scaling factor used to compensate for the α terms assigned to unity when determining the probability expressions. k can be modified empirically to bring the magnitudes of migratory populations in line with those observed in reality. Ideally k should incorporate reciprocal terms of α, which are generally functions of s'.

Currently, all probabilities are evaluated at both stations, and probabilities combined directly by multiplication. This conceivably introduces some approximation, since the flight distance will be an unknown function of the combined wind speeds at the two stations, and probably other unknowable factors. Further, the individual weighting of each of the probabilities might differ. At present using direct multiplication, all four probabilities are assigned equal favour. It might prove necessary to raise each probability to a (different) fractional

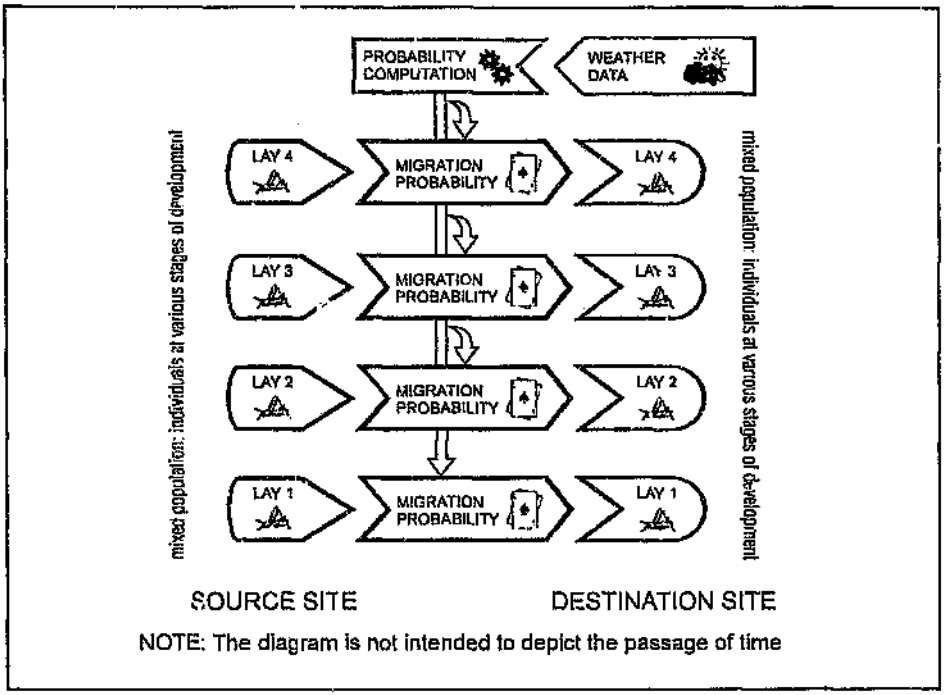


Fig. 3.12 Graphical representation of the migration engine application.

power to account for their different importance. This cannot be determined from the limited simulations performed so far.

However, the approximation is retained since there is no satisfactory method of determining exactly how the wind speeds and headings should be combined or averaged. If the sites are reasonably close, the wind speeds will not differ markedly, and the error so introduced will be conceivably small.

MIGRATION AND NON-MIGRATION MODELS

The simulation allows the user to enable or disable migration, either prior to commencing a simulation run, or at any time during a run. When migration is disabled, no insects in the sexually mature adult female stages are permitted to leave the site at which they arose, even if conditions conducive to migrations arise. If all emigrations are inhibited in this way, there can be no individuals available for immigration, and all movement between stations is suspended. This has no effect on the lifecycle models at those stations, except as far as reduced depletions of populations.

The ability to run otherwise identical simulations, both with and without migrations, allows the effects of the simulated migrations to be given careful scrutiny.

3.4 LIMITATIONS ON MODEL COMPLEXITY

A key decision point when developing the model, as outlined earlier, was how to decide when the model was "good enough". The top-down approach to model building facilitates a changeable goal, whereas a bottom-up approach requires that the ultimate goal be specified before model development begins. Having adopted the top-down approach, we are able to add refinements to the model, as they becomes necessary. At each revision, the question "Is the model good enough now?" can be asked, and then depending on the answer - obtained by whichever statistical test is deemed most appropriate - development can be continued, or suspended.

What should also be considered when evaluating if the model is good enough, is the possibility of the model, or a part thereof, being too good. Improvements in a certain section might make accuracy introduced into another section of the model redundant. Care should be taken not to create an unjustifiably unbalanced model, where some portions involve extraordinary degrees of detail and accuracy, and other areas involve the minimum of detail. In certain instances, such situations are justified, and no part of the model should be regarded as ultimately complete, while development is progressing on another portion.

A logical question to ask might be "What is wrong with a perfect model?" There are two immediately apparent failings of such a perfect model. The first being that it is too close a replication of reality, and as such doesn't really teach or explain anything. A simple model, when it works, demonstrates that it has captured the complete essence of the problem, and that the unnecessary detail has also been correctly identified. The second failing of an inordinately complicated (and thence hopefully accurate model) is that it requires excessive resources to operate. Resources here are measured in terms of computing power of the system on which the model executes, as well as the human factor involved in maintaining the model, providing it with input data, tuning the simulation parameters, and evaluating its output.

COMPUTER HARDWARE CONSIDERATIONS

When the original lifecycle model was transported to the PC environment, one of the prime arguments for this hardware platform was its universal accessibility to a large number of users. Currently there exist a wide range of computers, all of which claim to run MS-Windows software, but which differ markedly in their performance.

The model has thus been allowed to grow in complexity to stages where it performs impressively on state-of-the-art hardware, but is still acceptably useful on older and more common machines.

In order to achieve these performance goals, the model has been limited in complexity as follows:

- Two site simulation only. This adequately demonstrates the operation of the migration engine, and the principle behind simultaneous multiple site simulations. At the same time, it does not require exceptional information interchange. See Figure 3.13 below.

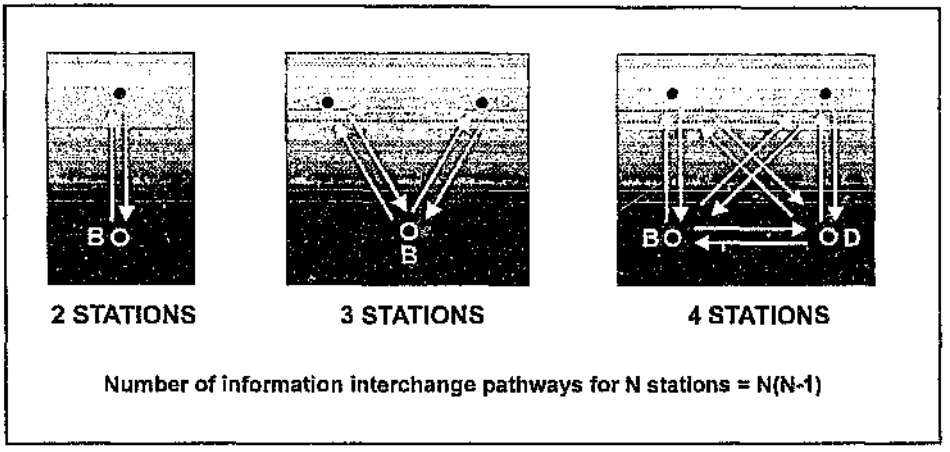


Fig. 3.13 Data interchange paths for increasing numbers of stations.

- Simulation time interval of one day (24 hours). Each simulation step requires a complete weather data record, that is, for all weather parameters, a measurement is required for each day/step. Increasing the temporal resolution of the simulation will require a proportionately larger weather data file. Additionally, since the weather data is converted into a workspace variable during the simulation, increasing the size of the file directly increases the required workspace (RAM requirement). Moreover, since the population record is updated and augmented at every simulation step, increasing the number of steps further increases the workspace. Loading the weather data file, and the pre-processing thereof requires a substantial portion of the simulation time. Keeping the weather data file to a minimum size is an important consideration in ensuring rapid simulation results.

- Linear approximations to degree-day contributions. The daily air and soil temperature fluctuations are assumed to vary sinusoidally. This is not strictly true, but there is no value in trying to fit a more accurate function to these data, since the data are not sampled at an adequate resolution for this purpose. Computing degree-day contributions from varying sinusoidal functions was deemed too computer intensive, and a linear (triangular) form curve is fitted to the temperature data. This facilitates rapid degree-day calculations, and is in line with methods recommended by Zalom *et al* (1983).
  Linear approximations to flight time as a function of day length. The same technique of linear (triangular) curve fitting used for the calculation of degree-day contribution is used to determine the expected time of flight. A

more accurate approximation to the assumed sinusoidal form was considered unnecessary, since the precise dependence of flight time on temperature is unknown anyway. Noise due to these uncertainties is anticipated to mask any errors due to the approximate curve fit.

- Omission of various lifecycle and migration parameters. Only the essential features of the migratory phenomena have been included. Less well understood, and scantily documented aspects have been omitted.

In some areas, the ultimate effects of applicable approximations were wholly unknown. If the omission of such approximations made no noticeable impact on the computing overheads, then the most accurate computation techniques were selected over the approximation.

One such example is the inter-station distance computation. These calculations take into account the convergence of lines of longitude as such lines tend towards the pole. Using a straight line approximation to longitude gave no detectable improvement in simulation time, and the potential error introduced - as would be reflected in locust population numbers - is unknown. This is estimated to be negligibly small, and in all likelihood will be obscured by other errors and approximations. However, since no computing power is wasted using the more accurate technique, it has been retained.

DETERMINING THE QUALITY OF THE MODEL

In the next chapter, the various aspects of how models can be tested are discussed. The availability of data suitable for testing the locust simulation model is described, as well as some statistical and non-statistical tools for interpreting the test results, and how these relate to the quality of the model.

Chapter 4 MODEL TESTING AND EVALUATION

4.1 MODEL TESTING CONCEPTS

VERIFICATION, VALIDATION AND SENSITIVITY

Wagner & Willers (1992) distinguish between validation testing and verification. Verification refers to the process of ensuring that the model or simulation executes as it should, whereas validation refers to the confirmation that the model is an accurate representation of the real system within its defined limits.

The importance of testing the validity of models is given great emphasis by McKinion (1992). He claims that a model can never be completely validated due to the possibility of non-uniqueness of the model. This is interpreted to mean that there may remain aspects of the model which cannot be probed by any available tests. Indeed this is certainly true of the locust simulation model, since it is not feasible to run simulations on every possible permutation of weather data and initial conditions. Thus there are potential behaviour patterns of the model that will only be revealed under certain chance conditions. After a level of confidence in the model has been established through validation testing, then the model can not only be used in a descriptive or explanatory capacity, but can be used to predict the behaviour of the modelled system in quantitative terms.

Models that perform adequately on some geographic locations may fail in others. In this case, adjustments of some parameter values may correct the problem. This procedure is known as calibrating the model. Shoemaker (1980), in Wagner and Willers (1992) , cautions that the field data used for validation is not necessarily good for calibration. Calibration is an acceptable tweaking option if the modified values of parameters do not differ markedly from their original theoretical values. The acceptable limits on this difference is a subjective quantity influenced by the conditions under which the parameters were determined (laboratory vs. field) and by the variance of the data.

Validation can only be performed once the model has been verified. Verification has been performed on a continuous basis throughout the model development. Easy verification is a consequence of OOPS programming, since each object can be tested quickly and thoroughly on its own. Errors are immediately apparent and there is no possibility of the source of errors being difficult to track down, or being obscured in large, unwieldy code structures.

The keystone of the simulation result analysis (validation) lies in comparing the simulated results to the actual outbreak records. Graphical representations of the simulation output provide a quick qualitative indication of the accuracy of the simulations. Various statistical tests are applied to give a quantitative indication of the closeness of the simulated results to the actual locust counts (locust control data).

Also important is sensitivity analysis. This is performed by varying the values of the input parameters (one - or sometimes more - at a time) and seeing the impact of these changes on the model behaviour (Worner, 1991; Wagner & Willers,

1992). Broadly speaking, there are two possible outcomes according to Wagner & Willers (1992). One is where small changes in a parameter value result in distinct changes in output, the model being sensitive with respect to the changing parameter. The other possibility is that even extensive changes in a parameter value will result in little or no difference in the model output. Shoemaker reports that sensitivity analysis is important in determining future research priorities.

In the case of a complex, multiple output phenomenon such as locust outbreaks, it is difficult to specify sensitivity. In the case of a simple model, sensitivity can be represented as a fractional change in output for a corresponding fractional change in the parameter of interest. For example, if the sensitivity to flight threshold temperature was being investigated, the sensitivity could be specified in units of locusts-per-degree-Celsius [$^{\circ}C^{-1}$]. Since in the locust model there is no easy way to represent the output as a single quantity, this single-va' representation is not possible. Furthermore, the sensitivity may be dependen the initial conditions (typical of a chaotic system), or the input data over the interval of observation. Thus the sensitivity analysis results are presented as a set of three-dimensional surface plots, in the x, t and y domains. x represents the parameter under scrutiny, t the time axis, and y the simulation output (number of swarms, or hoppers).

4.2 MODEL TESTING CRITERIA

REQUIREMENTS FOR TEST SIMULATIONS

The simulation model requires two types of data:

Firstly, weather data are required for each day of the desired simulation period. These data comprise maximum and minimum air and soil temperatures, rainfall, and wind speed and direction.

Secondly, in addition to the weather data, accurate locust count data are desirable. Although strictly this is unnecessary for performing simulations, it is required if the simulated results are to be meaningfully compared to reality.

Any missing data in the weather data sequence (i.e., periods where no measurements have been taken) cause the simulation to generate spurious results because missing entries are regarded as zeros by the data processing software. The errors depend on which measurements are missing:

- absent temperature values are translated into a null degree-day contribution, which will retard development of any stages experiencing those conditions.
- absent rainfall records may falsely contribute to premature death of certain egg stages due to dehydration, or false inhibition of development. A lack of rainfall records is thus interpreted as a prolonged and possibly unrealistic drought. In practice, rainfall records are good, and require very little interpolation.

- absent wind speed values result in the inhibition of migration, while missing wind bearing values - in the presence of valid numeric wind speed data - will cause migrations to be simulated to false destinations.

For these reasons it is imperative that continuous weather data be available in order to test the model accurately. In certain circumstances, where the missing data spans in the data are small and sparsely distributed, it is possible to insert estimated values to fill blank fields.

Weather data has been made available by the Agromet section of the ARC and also the SAWB. These data provide records of maximum and minimum air and soil temperatures, and rainfall records for a limited number of weather stations, and for limited time periods. The South African Weather Bureau (SAWB) has made available records of wind data for a limited number of stations and limited periods of time.

There are two formats of recorded locust data available:

1. Recorded data catalogued by latitude & longitude co-ordinates

   These data are available for 1994 and the first half of 1995. Most of this data is available only on paper, and requires entry into some data management program to be useable. This is a time-consuming process, and little - if any - of the processed data is available at the time of writing. Large quantities of the data are available in unprocessed form, but the lack of accurate location entries, and suspected errors in the dates make it of little value.

2. Recorded data catalogued by magisterial district.

   These records convey information over a much longer time period, from 1985 to 1992. These data are less specific, in that they only distinguish between hopper bands, and adult swarms sighted. Location is specified only by magisterial district name.

These two formats are sourced from the same original data set: the DRC survey "yellow forms" distributed to farmers throughout the Karoo.These forms collect detailed data on the magisterial district, farm name, latitude, longitude, and date of sighting, as well as the particular stage of development reached by the observed insects.

While the more complete locust count data (co-ordinate data) are more attractive from the point of covering the largest number of sites, the time period which they span is very restrictive. No wind or soil temperature data are available for those years. Furthermore, the accuracy of the dates is suspect, and the lack of *accurate* longitude and latitude fields makes it difficult to use (although work performed by Margaret Keiser of the PPRI should do much to solve the latter problem). The second set of locust data (based on magisterial district) remains the only count data corresponding to those years for which complete weather data are available. A substantial portion of that data is readily available in spreadsheet format.

It must be emphasised that these data represent quantities of locusts controlled (killed), and these are catalogued by magisterial district. Therefore, the regions do not coincide precisely with the points of weather data collection, nor do they accurately indicate the magnitudes of actual outbreaks.

Incomplete locust count data does not have any impact on the accuracy of the simulation. It does have an impact on the interpretation of the simulation results; a simulated outbreak might be interpreted to be spurious, when the real cause of the anomaly is the failure to record an actual outbreak.

SELECTION OF TEST SITES

The range of available potential weather data is summarised as a Gantt chart in Figure 4.1, overleaf. When selecting sites for test simulations it must be kept in mind that the simulation has to be performed at two sites simultaneously, otherwise migration has no meaning. The weather data for both these sites must span the same range of dates, which limits the simulation period to the shorter of the two data records. This is more than a case of meaningfulness - the simulation will be halted prematurely if one site runs out of weather data before the other. Lifecycle simulations can be performed with no migration for single sites by allocating fictitious weather data to one of the sites. It must be remembered subsequently that the simulation for that site is meaningless. Also required for meaningful interpretation of the simulation results, are the actual counts of adult swarms and hopper bands.

With these conditions imposed, it becomes apparent that the only useful sites for migration simulations are: Douglas, Graaff-Reinet and Pofadder.

The rarity of soil temperatures has prompted investigation of some simulations that omit the use of those records. De Aar provides good spans of all other data, and has been included in the test set. The climate data available for De Aar contains no record of soil *minimum* temperatures. The soil minimum temperature data for De Aar have been derived from that site's soil maximum temperature record. The method used to derive these missing records is analogous to that described below for emulation of soil temperatures derived from air temperatures - details are provided later. The difference between these two computations lies mainly in the scaling factors and temperature offsets required.

At the time of testing, there were no sites where "good" test data were available in all respects excepting soil temperature. However, this kind of data may become available in future, so the tests performed at De Aar - the site using minimum soil temperatures derived from the soil maximum temperature records - will help to determine if the model can be made sufficiently accurate without using some (or possibly any) soil temperature records.

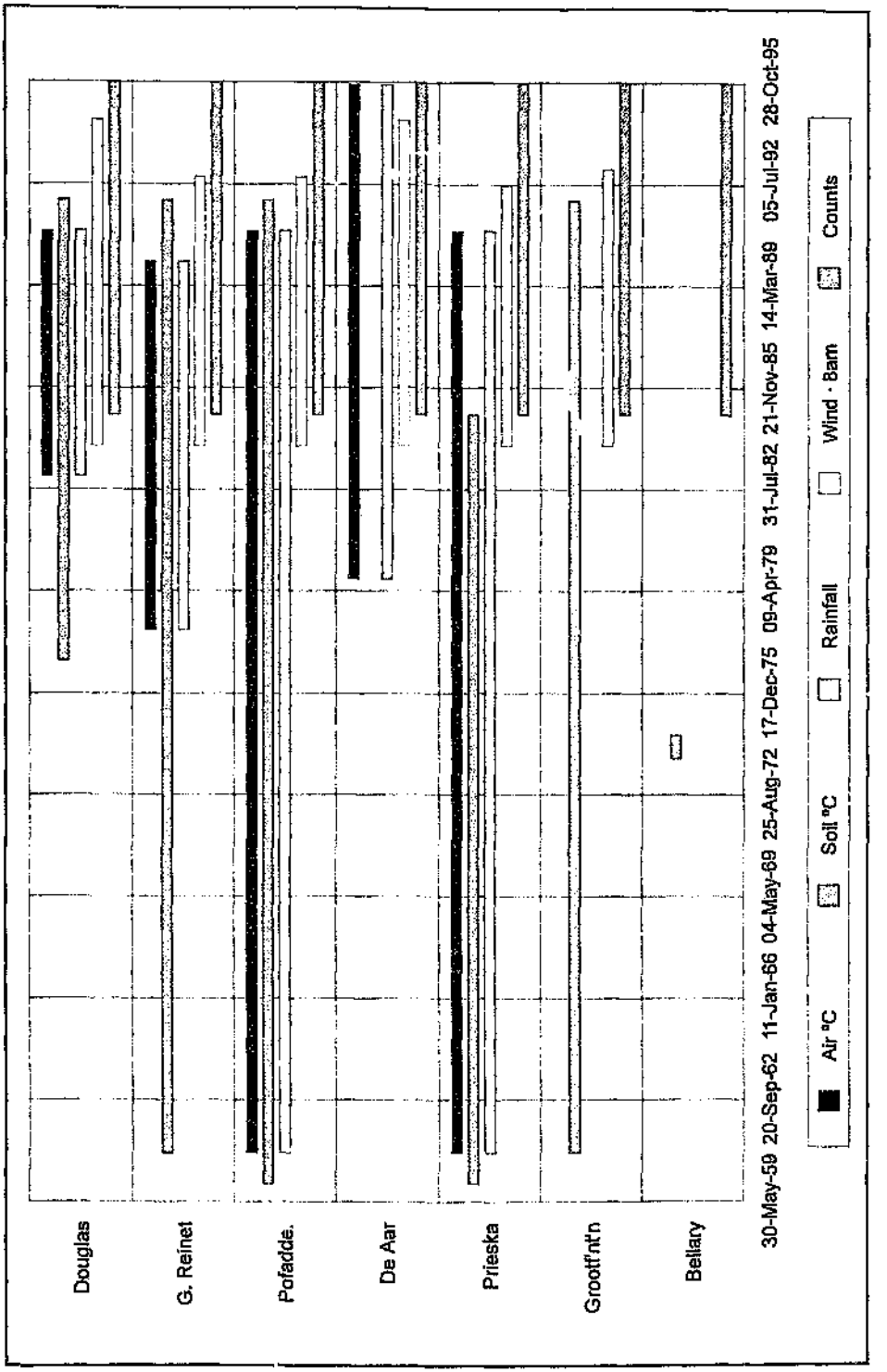


Fig. 4.1 Gantt chart. Available weather & locust count data for potential sites.

EMULATING SOIL TEMPERATURES

An ever-present problem throughout the development of the simulation model, was an inadequacy of climatic data. The only regularly obtainable measurements were those for maximum and minimum air temperature, and for rainfall. These were available on a daily basis for most potential modelling sites, and for almost all periods considered.

Much of the locust lifecycle hinges on the egg stage. In fact, if time spent in any stage is a reflection of the importance of that stage, then the egg stage is by far the most important. Being buried within their egg pod approximately 50mm below the soil surface, the eggs can be assumed to track the soil temperature closely. Thus soil temperature is a very desirable parameter to have available when monitoring the developmental conditions of the eggs. Unfortunately soil temperature measurements were extremely difficult to come by, even for a modest number of sites. There exist many methods for extrapolating soil temperatures at various (small) distances below the soil surface (McCann *et al* ,1992, Thunholm, 1990, Matthée, 1961), but these often pose more questions than they answer. These techniques invariably require measurements of the soil moisture content, and additionally or alternatively, the soil composition. Other desirable factors to know are the type of groundcover present, as well as the percentage cloud-cover, from which solar radiation effects may be estimated. Measurements of many of these factors were unavailable for the sites of interest.

Attempts were made to find simple correlations between soil temperatures and air temperatures, where both these measurements were available to us. However, no obvious relationship could be found, and simple linear mappings proved to have a high margin of error, indicating the importance of the other factors listed in determining the soil temperatures. Note that in all cases "soil temperature" refers to measurements taken 50mm below the surface.

Both the models of McCann *et al* (1992) and Thunholm (1990) require environmental and meteorological information that is not currently in the data set specification to which the model is required to adhere. Therefore, while attractive as these models may seem, another approach must be found. Three approaches are possible:

1. Some intervals within the simulation period lack both soil minimum and soil maximum temperature records. Under these circumstances, soil temperatures have to be based on air temperatures

   By inspection of temperature-time plots of air- and soil maximum and minimum temperatures for other sites, the following general trend emerges:

   The difference between minimum soil and minimum air temperature is an approximately sinusoidal function, with the same long-period and phase as the maximum air and soil temperatures (Long period refers to the seasonal fluctuations, not the day-to-day oscillations).

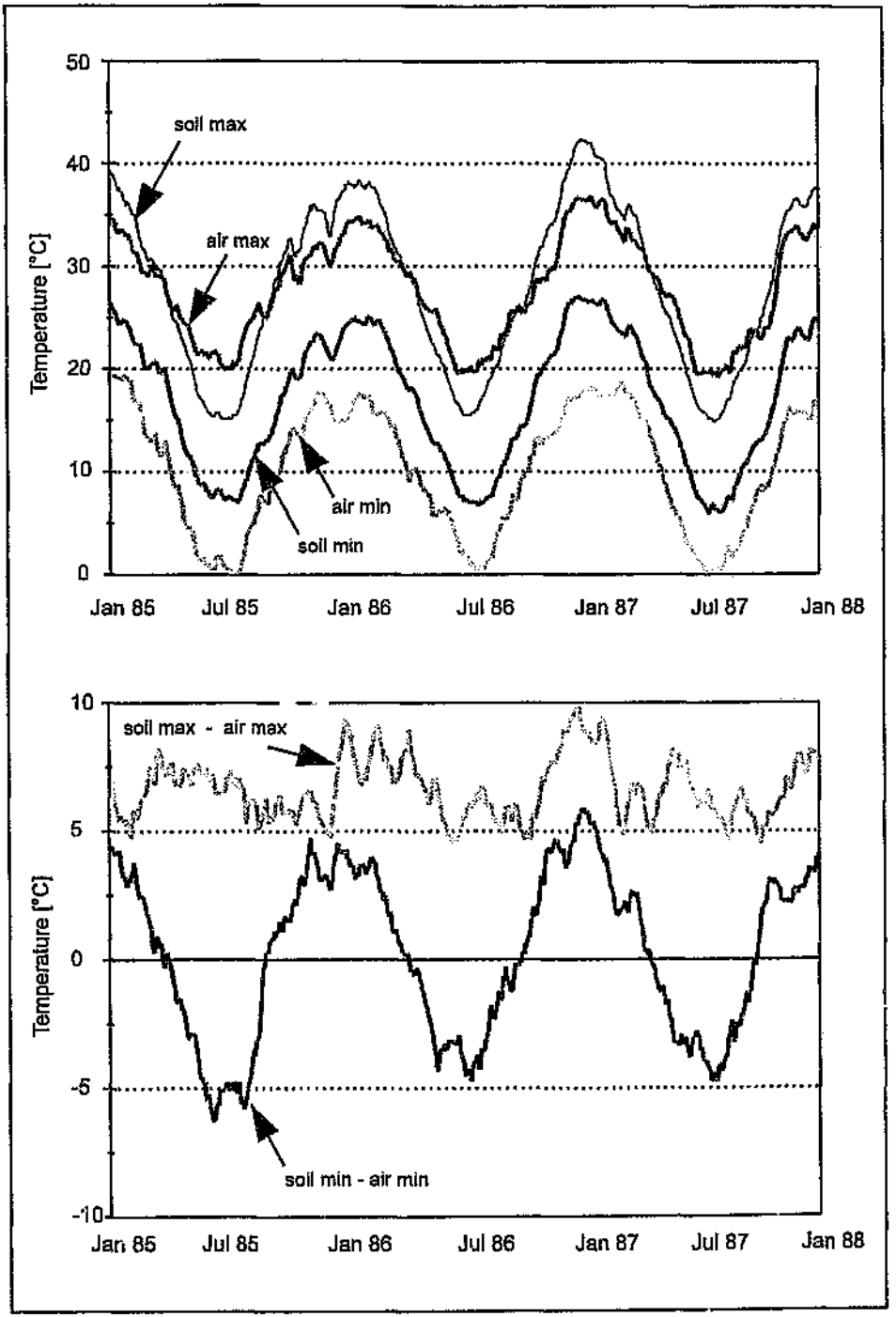


Fig. 4.2 Examples of soil and air temperature differences recorded at Douglas.

The long-period amplitude (or seasonal temperature range) in the air is smaller than in the soil (an unexpected observation). The consequence of this is that in summer the ($^{o}C_{soil}$ - $^{o}C_{air}$) difference for minima is positive, whereas in winter the difference is negative. Very generally, in summer the soil temperature tends to be higher than the air temperature; in winter the soil temperature tends to be virtually the same as the air temperature. Some of these trends are shown in Figure 4.2 for Douglas by way of example.

A rough estimate of the soil temperature can be made by smoothing the air temperature curve, and then adjusting the long-period amplitude and offset. The amplitude adjustment and offset adjustment is determined by inspection of records for which soil temperatures are available. The offset is readily calculated by finding the difference between the mean values of soil and air temperature ($^{o}\overline{C}_{soil}$ - $^{o}\overline{C}_{air}$) using the whole span of available records.

The y-axis shift coefficient should ideally incorporate a seasonal variation too, since ($^{o}C_{soil,\,max}$ - $^{o}C_{soil,\,min}$) is larger in summer than in winter (refer to Figure 4.3 later). Such a function could be synthesised to be of the form:

$$\Delta^{o}C = Y\cos\{\omega t - \phi\}$$

where ω is the angular frequency, $2\pi/365{,}25$ days^{-1}, and t is the number of days since January 1st. ϕ is an optional phase offset, to correct for mid-summer not falling on January 1st. $\phi \approx 0{,}025$. This corresponds to 9 days (December 22nd to December 31st) expressed as a fraction of the total number of days per year.

However, these finer points are better suited to work on soil temperature modelling, and will not be discussed in detail. Nor have they been incorporated into the temperature records used to test the locust simulation model.

Although this technique produces temperature curves that appear reasonably good, the alternative method - outlined below - has been used exclusively. This method, which is easier to implement, and achieves much the same result, involves seasonal average differences.

B. By looking at periods where all temperatures do exist, it is possible to calculate the mean difference between any two temperature records, over one season. It is important to limit the averaging window to a season or similarly small period because the mean differences change depending on the time of year. Once the seasonal mean difference is known, it can be used to compute unknown soil temperatures by assuming that the seasonal mean difference does not change from year to year.

When performing the calculations the seasonal means are computed for 30-day sliding windows. In other words, the "season" is regarded as a thirty day period centred on the day for which a value is to be predicted.

In this way, soil minimum temperatures can be estimated from air minimum temperatures, and soil maximum temperatures can be estimated from air maximum temperatures. This approach is easy to apply to any data within standard spreadsheets, and if the spreadsheet cells contain the appropriate formulæ, rather than values, the spreadsheet automatically keeps itself updated as data are added or changed.

C. For De Aar, no soil minimum temperatures exist. There are good records of soil maximum temperatures. Under these conditions, basing the soil minimum temperature on the soil maximum temperatures is more desirable, because these two show smaller deviations from each other than either do from the air temperatures.

Except during mid-summer, the soil temperatures (maxima and minima) generally lie between the air maxima and minima, due to the higher specific heat capacity of the soil. The soil and air maximum and minimum temperatures for a short span of three years is shown for Douglas in Figure 4.3.

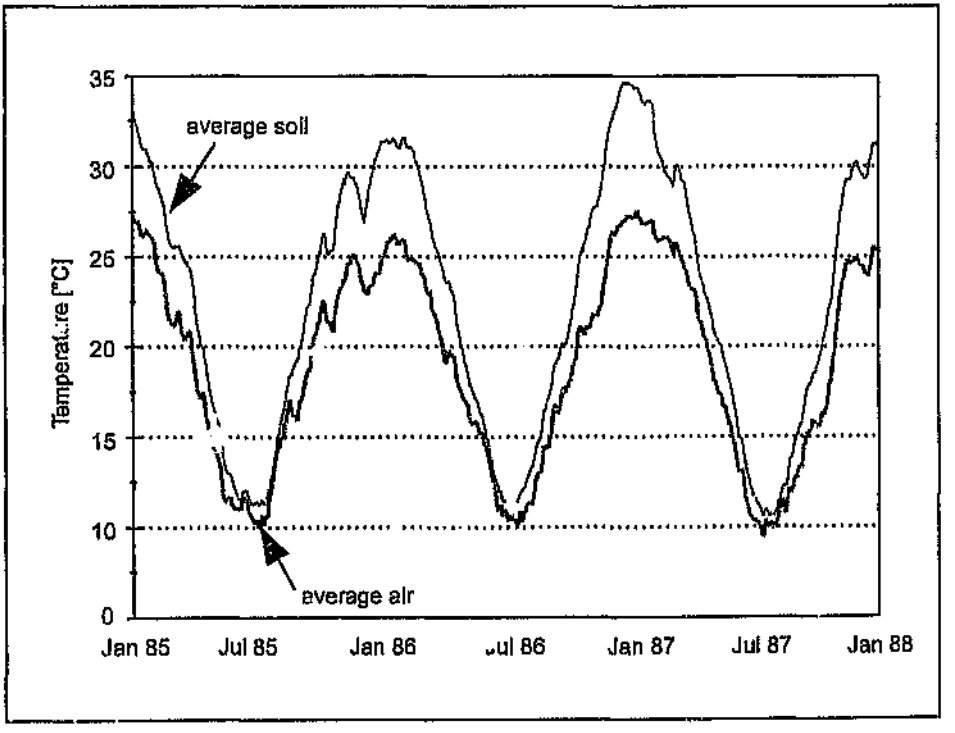


Fig. 4.3 Example of air and soil maxima and minima recorded at Douglas.

The soil minimum temperature is estimated from the soil maximum temperature by using the technique of mean seasonal differences again. However, since no soil minimum temperatures exist for De Aar at all, there is no easy way to evaluate a typical mean seasonal difference between soil maximum and minimum temperature. The seasonal mean has thus been based on that computed for Graaff-Reinet. This site is the closest to De Aar of all available sites, and it shares a similar climate (within the limits of available sites).

Thus, for a 30-day sliding window, which moves throughout the entire five-year span of data, the mean difference between Graaff-Reinet's soil maximum and its soil minimum is computed. This difference is used to find the estimated soil minimum at De Aar, for the corresponding period.

For evaluation purposes, the technique was used to generate soil temperatures - based on air temperatures - for those sites where soil temperatures *were* present. These were then graphically compared to the recorded soil temperatures, and the technique was found to provide much better tracking than simple equations relating soil to air temperatures. More comprehensive data sets, hopefully available in future, will obviate the need for these techniques for temperature estimation.

RELIABILITY OF DATA FOR CHOSEN SITES

The sites and time intervals chosen for the test simulations and model evaluation comprise the most complete sets of data available at the time of writing. The weather data for the four sites chosen (most especially De Aar) contains missing spans that vary from sporadic absences of single entries for various temperature fields to whole months or more of missing entries! Douglas is the only site for which there are continuous records of all temperatures and rainfall.

In such cases of data absence, some estimated values have to be used, because the simulation fails to execute correctly if it encounters missing entries. Various techniques and approaches have been adopted to compensate for lack of actual measurements, these being dependent on the magnitude of the missing data span. For the temperature curves, in essence, the aim of the data replacement - and measure of the quality of the technique used - involves plotting the recorded and replaced data as a function of time, and noting the smoothness of the curve. The lack of obvious discontinuities, or noticeable deviations from the approximately sinusoidal temperature fluctuations, indicate a poor data estimation technique.

- Where the missing datum is a single field for a particular day - such as one of air maximum or soil maximum or air minimum or soil minimum, but where the other three entries are present - the entire site data is searched for days where those three fields match most closely. That searched-out day's corresponding data is used to replace the missing field.

- If multiple fields are missing on a single day, an intermediate value for each measurement is chosen, based on the measurement for the preceding- and subsequent day. This approach is expanded for sequences up to several days of missing data. If two, or three, or four consecutive days are missing records, then a sequence is constructed that smoothly links the value recorded on the day before the missing data span to that on the day after the span. This is not based on any documented mathematical computations or formulæ; a simple average, or mentally calculated estimate of the average is used. This is considered acceptable since there is no available absolutely accurate method of predicting the missing values.

- When runs of consecutive days are found with no records for any fields, a small window on either side of the missing data span is examined for a trend: either falling temperatures, or rising temperatures. That trend is continued to the mid-point of the span, from both the start and the end of the span. Again, no elegant or rigorous mathematical treatment is involved, but if the followed trends lead to a mid-point value that seems subjectively unreasonable, then the slopes of the trends may be modified to achieve a more acceptable midpoint. Instances of very large missing data spans are rare in the time ranges selected because their occurrence is considered reason enough for discarding that site or time period.

- Large missing data spans in all four data fields (one month or longer) are filled using weather data from previous or subsequent years, for the same period. The year chosen as the 'donor' is selected by subjectively evaluating the step size introduced at the transition boundaries. The selected year is that which minimises this discontinuity.

- Missing rainfall values are almost always replaced by a zero reading. Rainfall in the Karoo is sporadic and seldom occurs on successive days. The likelihood of rain falling on the particular day when no data was recorded is considered low enough to discount. The exception to this is when missing rainfall entries arise simultaneously with noticeable drops in air and soil temperatures. This suggests the occurrence of rain. Rainfall values that typify those of the season (inspection of previous or subsequent years) are inserted in locations corresponding to the discontinuities in the temperature records. There are very few occurrences of missing rainfall data (Appendix D).

- Missing data in the wind speed record are always replaced by zeros. If any fields exist where there are wind speed measurements but no directional information, the speed value is zeroed. Basing a migration event on a false wind bearing can introduce larger errors that the suppression of that event. A direction measurement in the absence of a speed measurement is not a serious problem, since the speed will be interpreted to be zero, and result in a low probability of migration. There are further comments on the effect of this interpretation under the section "Known flaws in the simulation model".

Much more accurate wind data have recently become available from the S.A. Weather Bureau, and use of this should preclude the need for such estimation in that field. More on this is to be found in the section "Recommendations" in Chapter 6.

The large missing data fields are highlighted in Appendix D..

The mean seasonal difference is always computed for a 30 day window at the donor site, centred on the date of the missing datum.

It is not possible to enumerate every single instance where real data is missing, since there are a great many small missing data spans that occur during the chosen simulation period. However, when analysing the simulation results, if anomalies coincide with these data adjustments, then these are highlighted.

The original Quattro-pro™ .WB1 spreadsheet file in which the simulation data was assimilated has been colour coded to indicate *all* entries that are not measured values (i.e., synthesised or substituted entries). The reader may wish to refer to the disc file when regarding the simulation results. A paper printout of this file cannot be included in this document due to the enormous size of the spreadsheet (over one hundred thousand entries).

TEMPERATURE AND RAINFALL MEASUREMENT ACCURACY

The daily maximum and minimum air and soil temperatures - where available - are recorded to an accuracy of 0,1°C. Rainfall is measured and recorded to an accuracy of 0,1mm. All of these measurements are considerably more accurate than necessary, and no errors are expected due to failings in the precision thereof. A point to note when inspecting the simulation input files (spreadsheet data files) is that the numerical records represent a decimal fraction, although the decimal point is not recorded. In the case of temperature, rainfall and wind speed records, the least significant digit is regarded as the fractional part. That is, an entry of "123" is interpreted to be 12,3 degrees, millimetres or knots or whatever units are appropriate.

SITE PERMUTATIONS

The number of two-site simulations possible with N available sites is given by

$$\text{simulations} = \sum_{n=1}^{N}(n-1) = \tfrac{1}{2}N(N-1)$$

Hence, with four sites, there are only six possible permutations. The simulation based on Pofadder, De Aar and Douglas can be of longer duration, since these sites have longer air temperature records than Graaff-Reinet. The data spans are given, with the element limiting the span given in brackets.

Douglas	1 Jan 1985 (counts)	···	31 Dec 1990 (air °C, R^{mm})
Graaff-Reinet	1 Jan 1985 (counts)	···	31 Dec 1989 (air °C, R^{mm})
Pofadder	1 Jan 1985 (counts)	···	31 Dec 1990 (air °C, R^{mm})
De Aar	1 Jan 1985 (counts)	···	31 May 1992 (counts)

Also provided is a graphical representation of all the recorded outbreaks for the potential four sites, Figure 4.4, overleaf. Inspection of this bar chart will show that outbreaks have occurred more-or-less throughout the period at all four sites, except De Aar, where there was an obvious recession during 1989 and 1991. Note that Douglas was the only one of the four sites to show any record of outbreaks during the whole of 1991.

The apparent monthly periodicity of the outbreaks revealed by the bar chart is due to the count records being monthly cumulative totals.

This spread of outbreaks and recession periods presents a superbly diverse set of scenarios for testing the simulation model. It makes sense to run simulations over the whole January 1985 - December 1990 interval, so that the reliability of simulated outbreaks can be gauged. This includes possible false alarms, which can only be investigated properly in times of outbreak recessions.

In earlier non-documented tests done on the model, test runs were initiated from September 1st, in order to catch the beginning of the locust season. Initial conditions were based on quiescent egg reserves with no individuals in other stages. However, starting a simulation so close to the first "anticipated" outbreak (November '85) may cause the simulation to omit consideration of events during the previous summer (January '85) which might be contributory factors in initiating outbreaks the following season.

INITIAL CONDITIONS & OUTBREAKS IN 1985

When running any test simulation starting in January 1985, there were always found to be simulated outbreaks coinciding with the rains at the beginning of February 1985, and on March 10, and March 15. These simulated outbreaks are not indicated to have occurred in the actual records (refer to Figure 4.4).

In fact no outbreaks are recorded before April 1985 anywhere in the region, although this is possibly due to poor record keeping and failure to observe any swarm events.

Further, the simulation records insect or swarm increases, whereas the actual data records numbers controlled. This might - in many circumstances - account for discrepancies.

During April of 1985, swarm events were recorded at the following magisterial districts only:

- Koffiefontein
- Pertrusberg
- Vosburg
- Carnarvon
- Britstown
- Prieska.

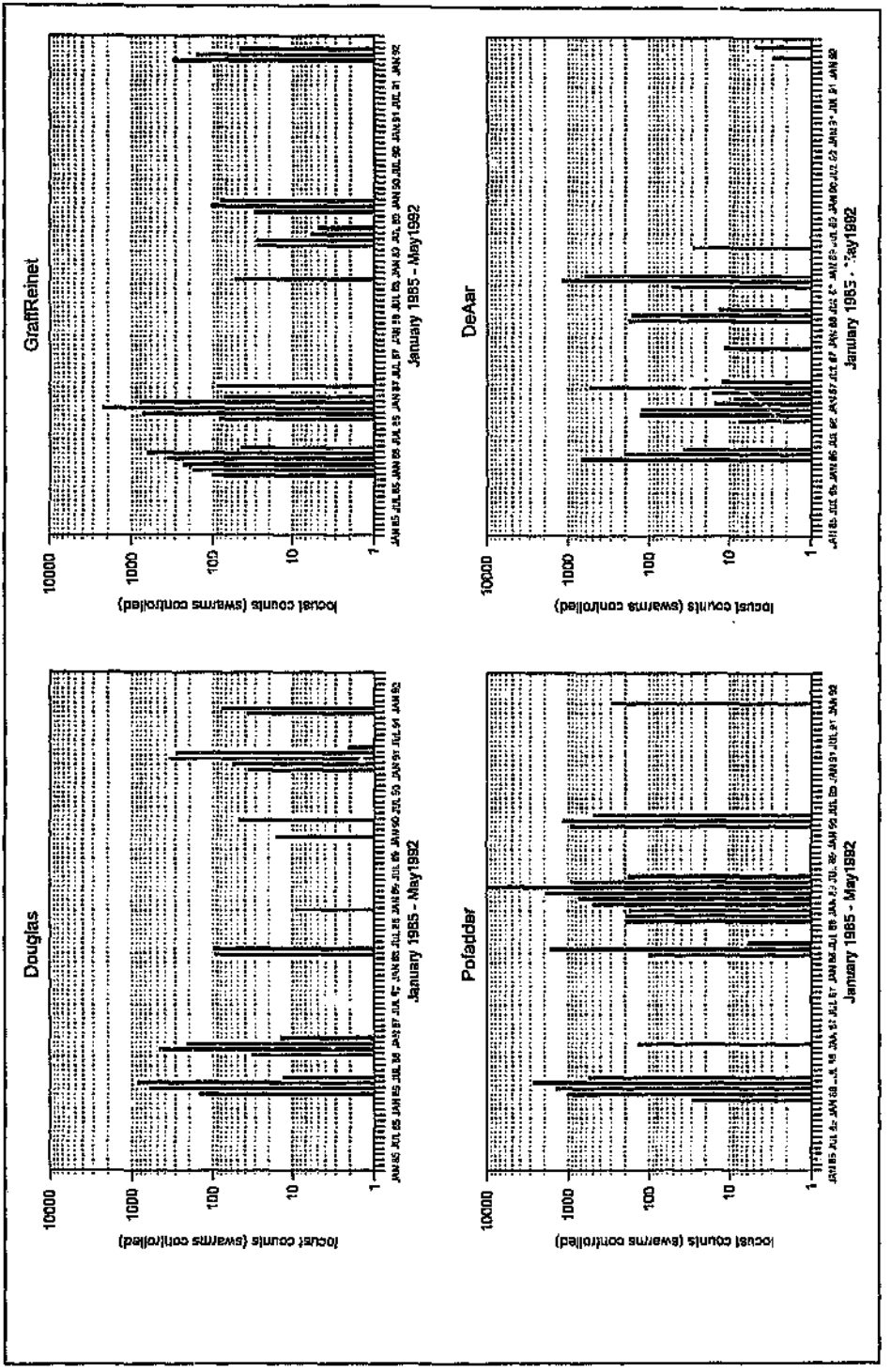


Fig. 4.4 All outbreaks recorded for the period 1985-1992 at sites of interest.

The reader is referred to the map (Figure 4.5) for the positions of the towns associated with these districts. Once again, a note should be made that the outbreaks are classified according to magisterial district, whereas weather data are associated with the nearest town. Unfortunately, the resolution of the weather-record grid does not favour a closer match of these two data sets.

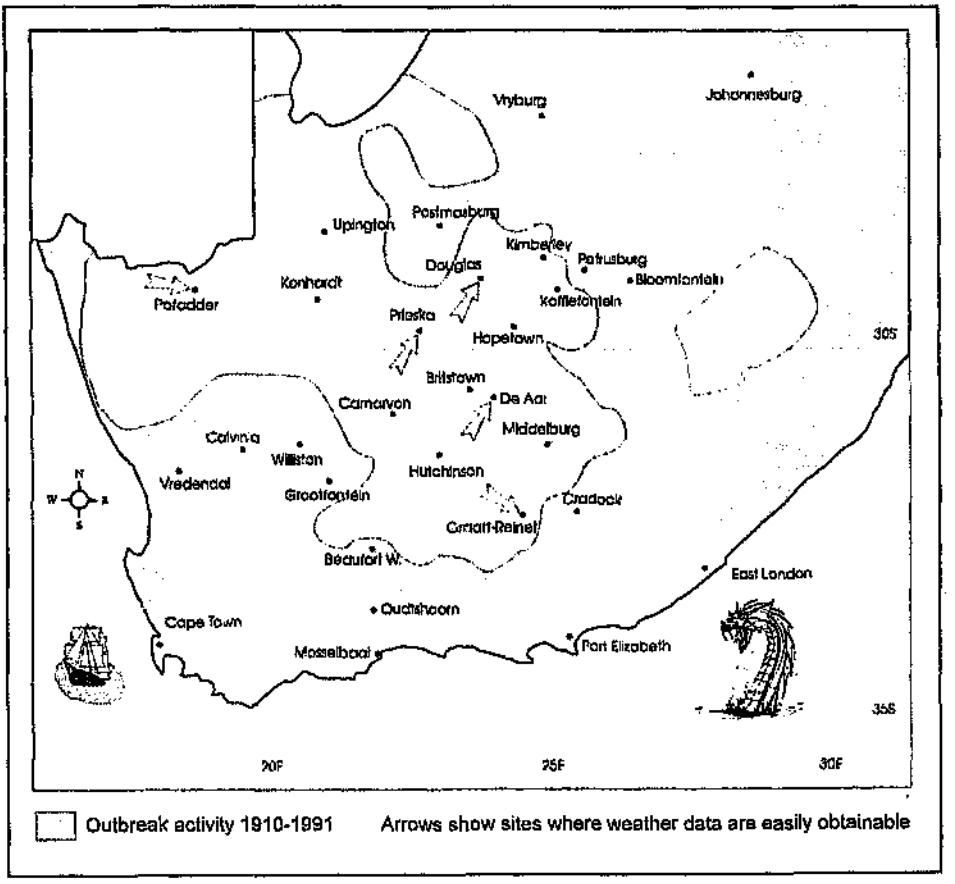


Fig. 4.5 South Africa: locations of sites of interest. Locust count data are catalogued according to magisterial district, but weather data are catalogued according to town.

A possible explanation for the paucity of outbreaks during 1985 lies in there being no viable eggs at any of the modelled regions, and migratory swarms may have flown in at a later time. This secenario cannot be simulated with the present model's available data since there is no weather data available for the above six sites where early 1985 outbreaks occurred, so they cannot be included in a simulation. Further, it is not possible to run a simulation with no initial insects or viable egg reserves, such as would be required for the sites where spurious February and March outbreaks are noted to have been simulated. Any such simulation would be completely static, showing no population fluctuations at all.

Clearly the problem of the false early-1985 simulations lies in the initial conditions. The only way to preclude these false simulated outbreaks was to specify no egg bed reserves in any of the four test sites (Douglas, Graaff-Reinet, Pofadder, De Aar).

Then, since the initial conditions reflect no eggs, simulations are started at the first observed outbreak, i.e., March 1986, and using observed adult swarms and hopper bands as initial conditions. This is an inconvenient date and it would be preferable to start simulations at the beginning of the year, allowing day '1' of the simulation to be assigned to January 01. Starting the simulation in 1986 obviously prevents simulating any part of 1985, a minor drawback. Initial conditions for the 1986-onwards simulation then become ones with no egg reserves, but realistic adult and hopper numbers.

In some ways this is better than using initial conditions involving egg numbers, because there are no records or even adequate estimates of egg reserves in the soil. However, it is also an unrealistic approach, since eggs are known to be very resistant to drought and can endure long unfavourable periods; it is extremely unlikely that there are in fact no eggbeds at all. Thus there does remain the problem of deciding whether to ignore the eggs in initial conditions, or to use estimated numbers that largely have no observations to support them.

START AND END DATES OF GENERAL TEST SIMULATIONS

The following six simulation runs are thus possible:

Douglas ⇔ Graaff-Reinet	1 Mar 1986 ··· 31 Dec 1989	1432 days	4 years
Douglas ⇔ Pofadder	1 Mar 1986 ··· 31 Dec 1990	1797 days	5years
Douglas ⇔ De Aar	1 Mar 1986 ··· 31 Dec 1990	1797 days	5 years
Graaff-Reinet ⇔ Pofadder	1 Mar 1986 ··· 31 Dec 1989	1432 days	4 years
Graaff-Reinet ⇔ De Aar	1 Mar 1986 ··· 31 Dec 1989	1432 days	4 years
Pofadder ⇔ De Aar	1 Mar 1986 ··· 31 Dec 1990	1797 days	5 years

There is no conceptual distinction between site A and/or site B in the simulation model, hence a simulation A⇔ B is the same as B ⇔ A.

Loading two data files of different length causes problems with the simulation's data processing modules. For instance, the data file for Douglas represents 1797 days whereas that for Graaff-Reinet only represents 1432 days. In order to overcome this difficulty, the sites with longer records (Douglas, Pofadder & De Aar) each have had their associated data files truncated. Thus test simulations run until the end of 1989.

4.3 SIMULATION RUN OUTPUT PROCESSING

Graphical representations of the results are provided in Appendix A, both for the non-migratory model, and for the model with migration enabled. A five year simulation is quite within the capabilities of the model, and completes in under ten minutes, even on an older 486-based computer.

Test runs were performed in single executions, i.e., the entire four- or five year span in a single run. The simulation output was recorded to a WK1 spreadsheet file, this file was loaded into Quattro-Pro™ version 5.0 for Windows. Quattro-Pro was used to create the graphs shown in this document, since it provides superior tools for manipulating and annotating graphs. The rank correlation results quoted are generated directly by the simulation model simply by specifying the "statistics" option from the model toolbox. The nature and interpretation of these statistics is described later in the section 4.4 "A statistical approach".

The simulation output, in its most rudimentary form, consists of tabulated lists. These lists represent the numerical magnitudes of population stages for all days of the simulation. It is possible to monitor any population stage, from eggs through to sexually mature adults. Examples of lifecycle stages are diapause eggs, first level quiescent eggs, instars, adult males, mature females of 3rd lay stage, and so on. Selection of which population stage to monitor is achieved by modifying the stages' names listed in the monitor software module. Details of these procedures are provided in the software technical reference accompanying this document as appendices (Appendix E through G).

Instars are not distinguishable into individual first, second, third etc., nymph stages. This is a consequence of the way in which the software code has been written and does not imply that these stages are biologically indistinguishable. There are no important differences between the different nymph stages, as far as the simulation model is concerned. Therefore the same rules may be applied to all nymph stages, allowing concatenation of these into a single software object.

The choice of which lifecycle stage to use for model evaluation is completely determined by the availability of actual recorded data. Tests on egg stages are not meaningful if no records of actual egg numbers are available. To this end, the evaluation has concentrated on hopper band counts, and adult swarm counts.

MODEL TEST CONDITIONS

The model testing was performed in two phases. Absolute accuracy is tested by executing multiple simulation runs, on the same weather data, but trimming the migration parameters where necessary in an attempt to make the simulated results fit the recorded data as closely as possible. Initially the migration engine is switched off, disallowing migration. On the second run for the same weather data, migration is enabled, and the migratory results are compared directly to the non-migratory results. If any fine tuning of model parameters are required, both migratory and non-migratory simulations have to be redone before direct comparison of the results may be done - unless, of course, the changing parameter relates specifically to the migration model.

Sensitivity testing is performed by executing multiple migratory simulation runs, but trimming the migration parameters to see how changes in these affect the overall simulation results. There is no direct comparison of migratory results with non-migratory results.

Note that there is nothing to be gained from running repetitive simulations on the same data, using unchanged model parameters. The model is wholly deterministic and thus will produce the same, identical result each time it is executed. Applying measurements of variance to the outcome is irrelevant; if a parameter is edited, a different model now exists, which cannot be directly compared with former models in measurements of this sort.

A potential cause for concern is that the lifecycle model and its associated parameters have not yet been tested fully. Detailed testing of the lifecycle parameters is beyond the scope of this report, but it must be noted that errors that exist within the lifecycle model can mask or exacerbate any flaws that exist in the migration model.

A feature of the complete migratory model is the facility to inhibit all migrations, even when suitable conditions for migration prevail. In this way, further testing of the lifecycle model alone is easily done.

The complete migratory model is a complicated entity. Lifecycle parameters combined with migratory parameters allow for an enormous number of permutations. Only brief testing has been done to verify that the introduction of some essential migratory modelling techniques leads to a significantly more or significantly less accurate simulation than that produced by the lifecycle model alone.

Migration and lifecycle parameter settings in effect during the simulation run are tabled in Appendix H.

4.4 A STATISTICAL APPROACH

A reasonably detailed description of the statistical approach is presented since the tests form an important component of the software created. Also, the locust population data presents special problems that merit highlighting, and correct interpretation of the statistical tests requires an understanding of how these tests operate.

RANDOM VARIABLES

The magnitude of the population of any particular lifecycle stage can be considered a random variable. Note that this is a statistical convention, and does not imply that the numeric value is truly random in the sense that random numbers are random. The term random variable refers to a mapping function whereby it is possible to map the outcomes of an event to a numerical quantity. In the case of the experimental data being numeric, the data itself can be considered the random variable (no explicit mapping is then required).

Since the actual population is known - from the count records, and the simulated (or predicted) population is also known - once the simulation run has been completed, it becomes apparent that there are two random variables.

It is convenient to consider only one lifecycle stage population at a time.

- Let the random variable associated with the simulated population be Y
- Let the random variable associated with the actual population be X

These assignments are purely arbitrary, and any variable name could have been used. These should not be categorised as dependent and independent variables. The nomenclature does not intend to imply that there is a relation (in the pure mathematical sense) between X and Y. These variables are wholly independent of each other, since there is no cause/effect mechanism, except in the broadest sense that both are linked to the weather. Notwithstanding, the actual count record is an unchanging record, whereas the simulated count record is variable, depending on the model parameters. The described statistical tests seek to determine a correlation between the two variables, but this does not imply that changes in one can cause changes in the other. Note that both X and Y are (discrete) functions of time, and thus are more completely denoted as X(t) and Y(t).

It is equally acceptable to regard these two random variables as a single bivariate random variable. In other words, instead of considering that there exist two random variables which are the outcomes of two independent processes X and Y, it is possible to regard them as the paired result of a single process - a bivariate random variable Z=(X;Y). This is more a matter of conceptual convenience under some circumstances and does not alter the analysis.

JOINT PROBABILITY FUNCTIONS (JPFs)

Consider now two random variables X and Y, the outcomes of two independent processes:

It can be shown that under conditions of a normal joint probability function for X and Y, the discrete-sample correlation coefficient between X and Y is given by

$$\rho_{discrete} = \frac{\Sigma_i (x_i - m_X)(y_i - m_Y)}{\{ \Sigma_i (x_i - m_X)^2 \ \Sigma_i (y_i - m_Y)^2 \}^{1/2}}$$

where $m_x = \frac{1}{N} \sum_{i=1}^{N} x_i$ the *expected* or *mean* value of X

and similarly for m_y

Note that this expression for ρ is a special case; it is a simplified form of the normalised co-variance of (X,Y) that applies when the joint probability function $p_{xy}(x,y)$ is assumed to be uniform.

Essentially what this expression says is that the correlation coefficient, rho, the special-case normalised co-variance, is a function of the difference between each and every x value and the mean of X, and the difference between each and every y value and the mean of Y.

The correlation coefficient refers to a linear correlation between the random variables X and Y. It is not so much a measure of how dependent X and Y are upon each other in a general sense, but is a measure of the *linearity* of that relationship. ρ^2 is called the coefficient of determination, and is a relative measure of how well a straight line fits the data (x_i, y_i)

ρ above is also known as Pearson's Product Moment.

It is tempting to accept this analysis approach, and utilise this ρ as a measure of the correlation between the predicted and recorded results. Such an approach has been used to analyse earlier models (Nailand & Hanrahan, 1993) with respect to rainfall. Unfortunately, this is an unsatisfactory approach - due to the fact that the unknown JPF describing (X,Y) is invariably assumed to be uniform. As stressed before, with these assumptions it is possible to simplify the normalised product moment to the familiar correlation coefficient as shown above (Stremmler, 1982).

The fact that the normalised product moment approximates a coefficient of correlation only when the JPF is of the required canonical form is seldom emphasised in texts (Neave & Worthington, 1988, Conover, 1971; Stremmler, 1982). The problem that arises is in the interpretation of ρ. ρ itself is a random variable, with an associated probability density function (PDF). That PDF is a function of the JPF of the paired data (X,Y). If the JPF of (X,Y) is unknown, then the PDF of ρ is unknown, and thus ρ is of no value as a test statistic, or in forming confidence intervals.

Inspection of the distributions of either the actual locust count data (Table 4.1 overleaf), or the predicted population data shows immediately that neither even vaguely resemble uniform nor Gaussian distributions, due to the preponderance of zeros.

For each site, the left column shows the magnitude of an outbreak, the corresponding right column shows the frequency of occurrence for that magnitude of outbreak. The data are not presented as a histogram, because the necessary probability-axis scaling causes the majority of the data to be "lost", these values being too small to be seen above the x-axis.

This PDF of undetermined nature does not altogether preclude using the product moment as a measure of correlation, since theoretically it is possible to evaluate the mathematical expression describing the distribution of the populations. Knowing that would facilitate an accurate evaluation of the moment, ρ. The difficulty arises in the complexity of calculating such a distribution function, which does not remain constant. The enormity of this task is reason enough to turn to distribution-free analysis tests.

TABLE 4.1 : Data distribution of locust outbreaks

Douglas		Graaff-Reinet		Pofadder		De Aar	
magnitude of outbreak ↓	probability of occurrence ↓						
0	99.315%	0	99.133%	0	99.133%	0	99.087%
9	0.046%	5	0.046%	6	0.046%	1	0.046%
13	0.046%	6	0.046%	30	0.091%	8	0.046%
14	0.046%	28	0.046%	102	0.046%	9	0.046%
16	0.046%	31	0.046%	139	0.046%	12	0.046%
32	0.046%	45	0.046%	179	0.046%	13	0.046%
35	0.046%	54	0.046%	187	0.046%	14	0.046%
47	0.046%	66	0.046%	199	0.046%	16	0.046%
90	0.046%	74	0.046%	203	0.046%	17	0.046%
94	0.046%	82	0.046%	503	0.046%	29	0.046%
135	0.046%	92	0.046%	578	0.046%	39	0.046%
141	0.046%	104	0.046%	762	0.046%	53	0.046%
203	0.046%	106	0.046%	952	0.046%	130	0.046%
443	0.046%	181	0.046%	1040	0.046%	135	0.046%
595	0.046%	214	0.046%	1455	0.046%	166	0.046%
848	0.046%	368	0.046%	1725	0.046%	187	0.046%
		674	0.046%	1984	0.046%	209	0.046%
		747	0.046%	2857	0.046%	552	0.046%
		822	0.046%	10396	0.046%	639	0.046%
		2281	0.046%			712	0.046%
						1239	0.046%

TESTS OF THE DISTRIBUTION-FREE TYPE

Statistical tests of the type described so far are known as parametric statistics. Typically, it is assumed that there exists some statistical model, that is, one or more random variables that define the process. Further, it is assumed that there is a joint probability which is completely specified. As outlined above, the joint probability, p_{xy}, is unknown in the case of the locust population distributions. Non-parametric statistical tests distinguish themselves by making no assumptions concerning the knowledge of p_{xy}, making them extremely attractive for the required analysis. Assumptions which are still made under non-parametric analyses are:

- The observations in the samples are independent
- The sample distributions are the same and continuous, although other knowledge of that distribution is not required. An important point is that the distributions are not required to be symmetric.
- The Y_i's may be shifted with respect to the X_i's. What this means is that the distribution of (Y_i - delta) is the same as the distribution of X_i. Delta is some unknown parameter. If delta = 0 then the distributions of X_i and Y_i are the

same. For the simulation, this conveniently accommodates scaling errors. Errors in the overall magnitude of the output will not induce a poor correlation statistic.

The second assumption cannot be irrefutably stated to be true. But in the case of a reasonably accurate simulation (the goodness of which may be estimated by graphical representation of the two variables X and Y), the distributions may be considered to be at least similar.

Several non-parametric or distribution-free tests exist (Kendall, 1962; Bradley, 1968; Conover, 1971; Hollander & Wolfe, 1973; Lehmann & D'Abrera, 1975). As the name implies, these statistical tests do not require that the data conform to any of the common canonical PDF forms.

It is not obvious, although it can be shown to be true (Neave & Worthington, 1988), that the Pearson's Product moment approximation to the correlation coefficient applies equally to the rank of the data. By ranking the data (ordering it from smallest to largest, and assigning an index to each element) we eliminate the clumsy distribution function associated with the original data: the rank indices can be analysed.

Intuitively, when considering rank indices, the distribution function must be uniform. The rank index runs from one through to N, where N corresponds to the number of samples. Then there exists a single rank index for each sample. Samples in the case of the simulation refers to the number of days over which the simulation was run.

As noted above, if the JPF of (X,Y) is unknown, then the PDF of ρ is unknown. However, since the rank indices have a uniform PDF, the ρ associated with them has a known, calculable PDF, and thus can be used as a test statistic.

Violations of this uniform distribution assumption occur when there exist tied data. Tied data refers to the cases when consecutive or non-consecutive samples of either the X_i or Y_i data or both are identical. Such datum points cannot be uniquely ranked, and a special approach is required. These special cases will be discussed later under the section "Dealing with tied data". Unfortunately, in the case of the locust simulation data, and the real locust count data, tied data is likely to occur frequently, because on many days there will be no locust outbreaks. The zero values cannot be ranked satisfactorily, although they represent extremely important information. If the simulation predicts no locusts, and the actual records show no locusts, then the simulation is performing accurately, and the numeric statistic must be able to reflect this.

SPEARMAN'S RANK CORRELATION

Spearman's rank correlation coefficient test is one of the simplest and most frequently used of the non-parametric distribution free tests. Its simplicity should not belie its usefulness.

Let X_i and Y_i be the two random variables under consideration.

X_i and Y_i are ranked from lowest value to highest value, and given indices from 1 to n, where n is the number of sample pairs. The ranks are conveniently denoted by $r(X_i)$ and $r(Y_i)$.

Note that the X and Y data are not re-ordered. Ranking does not involve 'moving' the numerical quantities, merely assigning an index to each value.

For each pair [X_i;Y_i], with corresponding [$r(X_i)$;$r(Y_i)$] the difference between the ranks is calculated: rank difference$_i$ = $d_i = r(Y_i) - r(X_i)$

Each rank pair difference is computed, giving n values. Note that the rank difference may be a negative number.

The sum of the squares of the rank differences is denoted
$D^2 = d_1^2 + d_2^2 + d_3^2 + ... + d_n^2$

It can be shown that (Neave & Worthington, 1988) the Spearman's rank correlation coefficient is given by:

$$r_s = 1 - \frac{6\ D^2}{n^3 - n}$$

This formula is identical to that obtained if the derivation of Pearson's Product Moment is applied to the ranks instead of the actual data values.

What this correlation coefficient computes is essentially a function of $d_1^2 + d_2^2 + d_3^2 + ... + d_n^2$. The only circumstance under which r_s is close to unity is when all the d_i values are small. In other words, if D^2 is small. Such a condition arises only if small X_i occur with small Y_i, and intermediate X_i occur with intermediate Y_i, and large X_i occur with large Y_i. Intuitively it can thus be seen that 'good' Spearman's rank coefficients are found when the peaks and valleys of the two X(t) and Y(t) curves coincide, regardless of the actual magnitudes of those turning points. Less obvious is the fact that shifts in the median of either of the two random variables does not make the coefficient less meaningful.

Spearman's correlation coefficient does not test for a specifically linear relationship between X_i and Y_i, as does Pearson's product moment. Any monotonic function (defined below) that maps X_i to Y_i in a way such that increasing X_i corresponds to increasing Y_i will show up as a strong positive correlation under the Spearman test, whereas the function is required to be specifically linear for such a positive correlation to show up under the product moment test.

A monotonic function is defined as follows:

If, for any sequence, $a_1, a_2, a_3, ... a_n$

$a_n \leq a_{n+1}$

or $a_n \geq a_{n+1}$

then the sequence is monotonic.

In other words, every number is larger or equal to its predecessor (non decreasing sequence), OR every number is smaller or equal to its predecessor (non increasing sequence).

KENDALL'S RANK CORRELATION COEFFICIENT

The Kendall's rank correlation is less well-know than Spearman's, and has some theoretical points in its favour. The most relevant being that

> Its null distribution converges to normality faster than in the case of Spearman's coefficient (Conover, 1971).

This is important with small sample sizes, and becomes progressively less relevant as the sample size increases, or as the confidence interval widens. Typically a 5% confidence interval is used. Neave and Worthington (1988) suggest that a sample size of 50 [days] is sufficient to bring the difference between Kendall's coefficient and Spearman's to an inconsequentially small value.

Kendall's approach is based on a comparison of all possible pairs (X;Y) with every other such pair (Kendall, 1962). The correlation coefficient, τ, is determined by number of discordant, denoted N_d, pairs of $(X_i;Y_i)$. When comparing pairs for instance $(X_i;Y_i)$ with $(X_j;Y_j)$, $(X_j;Y_j)$ is said to be discordant if either $\{(X_i<X_j) \text{ and } (Y_i>Y_j)\}$ or if $\{(X_i>X_j) \text{ and } (Y_i<Y_j)\}$

The formula developed by Kendall (Kendall, 1962), gives τ as a function of N_d

$$\tau = 1 - \frac{4\,N_d}{n^2-n}$$

From this it can be plainly seen that for τ to approximate unity (a good positive correlation), N_d must tend toward zero. That is, there must be very few discordant pairs in the data.

Intuitively this makes sense, since when comparing $(X_i;Y_i)$ with $(X_j;Y_j)$, small X_i occurring with small Y_i will result in concordant pairs, as will large X_i occurring with large Y_i. Many concordant pairs imply few discordant pairs, and hence a low value for N_d.

INTERPRETATION OF CORRELATION COEFFICIENTS

There are a number of matters concerning correlation and its interpretation that sometimes trap the unwary into misinterpreting the test results.

Firstly, it is not true that a strongly significant result from either of the two rank tests implies, or is implied by, a high correlation between the X and Y random variables. What the tests actually evaluate is whether or not the null hypothesis of zero correlation is consistent with the (X;Y) data. If n is large - as will frequently be the case with the simulation model results - a relatively small value of r_s or τ may be highly significant. Secondly, even if r_s or τ turn out to be zero, or close to zero, that does not necessarily imply that X and Y are unrelated. They may still be very strongly related, since r_s and τ are measuring the extent to which a monotonic relationship applies between X and Y (although not specifically linear, as in the

case of Pearson's product moment). If the data is closely related by some non-monotonic function or relation, it is conceivable that all three correlation coefficients yield a value approaching zero.

Thirdly, r_s and τ measure a tendency towards monotonicity, not just linearity. This is a consequence of all distribution-free analyses, not only Spearman's and Kendall's tests. None of the coefficients of correlation derived from such tests relate directly to the numerical data, but only indirectly via their ranks. Thus the test measure the tendency of X and Y to increase (or decrease) together, irrespective of whether they do so linearly or otherwise.

The importance of this is dependent on context, and in the case of the simulation model there is little cause for concern. It is extremely unlikely that the simulation will consistently generate a set of output data that is some non-linear monotonic function of the recorded quantities. Yet even if this were to be the case, the resulting high correlation coefficient would be cause for jubilation, since it would imply very good temporal correlation, although at the expense of accuracy of population magnitudes.

Generating a scatter-plot of the predicted and recorded data sets will reveal at a glance whether the relation between X and Y is tending towards linearity or some other monotonic function.

DEALING WITH TIED DATA

Since the procedures for computing the two rank correlation indices are different, different methods are adopted for dealing with tied data.

Neave and Worthington (1988) outline a procedure for accommodating ties in the Spearman test. This involves a modification of the ranking procedure so that tied data values are not given sequential rank indices (such as 5,6,7,8, for example). The tied data are all given the same rank index, which is computed as the arithmetic mean of the sequential indices - were those to have been used. For example, if four tied values existed that would have arbitrarily been assigned indices 5 through 8, the data are now all assigned a common index, given by
$(5+6+7+8) \div 4 = 6\frac{1}{2}$

If the general assumption - that most ties for the locust data will occur at zero values - is adhered to, then this has an important consequence as far as using average indices is concerned. Because all the data values are zero, the index range must originate from one, for no datum value can be smaller than zero. The average of the 'tied' indices must always be smaller than the largest index used in the calculation of the index mean, by definition of the calculation of means. Thus the averaged rank index applied to all tied data points will still be the smallest index in the series. If there are n tied data points, then the average rank will be $(1 + 2 + 3 + ... + n) \div n$. This can be expressed by the formula $\tilde{n} = \frac{1}{2}\{n+1\}$.

Thus the average tied index, $\tilde{n}$, varies considerably depending on the number of tied samples. The precise effects of using $\tilde{n}$ in place of sequential indices depends on the type of test being used, and how the indices are used to compute the correlation coefficient.

The method currently used by the simulation model's statistic tools is that of averaging tied index ranks. The program code written for the statistical analysis relies on certain conditions and assumptions. Tied data are only accounted for if they occur for sample values of zero. This assumption dramatically speeds up the analysis, and reduces code complexity, since it does not require each and every datum value to be compared to all others for equality.

The Kendall test, based on counting discordant pairs, requires a slightly different approach to dealing with ties. Kendall's suggestion is to order the ties in all possible ways and then average the value of the statistics thus obtained. In the case of the locust data, virtually the only instance where ties occur is when both the sample populations (predicted and actual) are zero. In this instance, reordering the ties will cause absolutely no change in the statistic - since the values are all zeroes. Hence this approach does not provide a particularly helpful solution. Neave & Worthington recommend that to account for ties, when counting discordant data pairs, a count value of ½ be attributed in any instance where either the X_i or the Y_i data exhibits a tie. This does not have exactly the same effect as applying the tied rank average to the Spearman test.

When incorporating the possibility of tied data into the Kendall's test - and evaluating this test with synthesised data - it was found that in the presence of ties, the correction factor was larger than the analogous correction for the Spearman's test. This difference can be explained by the nature of the correction when applied to the Kendall test (Kendall, 1962). The differences in the correlation coefficients that were noted were of the order of 1½ percent. This is considered negligible.

TWO TESTS BASED ON THE CUMULATIVE DISTRIBUTION FUNCTIONS (CDFs) OF TWO POPULATIONS

Two popular rank-correlation tests have been outlined. These distribution-free tests operate on the assumption that the PDFs of the sample populations are the same. Tests which give an estimate of the sameness of the PDFs are those of the Kolmogorov-Smirnov type, and the Mann-Whitney test (Conover, 1971; Hollander & Wolfe, 1973; Neave & Worthington, 1988).

MANN WHITNEY TESTS

A distribution-free test also based on ranking the $[X_i;Y_i]$ samples; the Mann-Whitney test, yields a so-called "U" value, which may be interpreted with the help of critical value tables. Comparison of the computed U value with the tabulated critical value shows whether or not the Nul Hypothesis, "H_0", - the conjecture that the samples X_i and Y_i are *not* significantly different - is true or false. Critical values may also be computed using the standard formula. (Neave & Worthington, 1988). A version of the Mann-Whitney test, was developed two years earlier by Wilcoxon. This Wilcoxon signed-rank test assumes equal sample sizes - a criterion not necessary for performing a Mann- Whitney test, making the latter useful for partial simulation runs.

Unfortunately, the Mann-Whitney test is sensitive to differences between the two sample populations means *or* medians, but does not show up differences of other types, such as differences of variance.(Connover, 1971). In fact, the Mann-Whitney test and the Wilcoxon variant both suffer a reduction in power when the PDFs differ in both mean *and* spread. Further, the Mann-Whitney test can only tolerate a moderate number of ties, something which cannot be guaranteed with the output from the locust simulation.

THE KOLMOGOROV-SMIRNOV TEST

The Kolmogorov-Smirnov test is the most commonly used, and the simplest, of several distribution-free tests which compare two sample cumulative distribution functions (CDFs) in order to test for differences of *any* kind between the distributions of the populations from which the samples have been taken. (Neave & Worthington, 1988).

The Kolmogorov-Smirnov test statistic computes the maximum vertical difference between the two cumulative distribution functions. The CDF for a sample of size n is a step function which rises by 1/n at each observation, always starting at zero and ending at unity. A hypothetical example of two CDFs is shown overleaf in Figure 4.6. The maximum difference, D, depends only on the order of the sample values of the two populations, and not on their actual values.

Standard texts on statistical tests include critical value tables for D. More commonly, critical values for D* are given, where

$$D^* = n_A.n_B.D$$

since D* takes on integer values; it is easier to find the maximum difference between integers than fractions, when the test is done by inspection.

When interpreting D*, large values of D* will cause H_0 to be rejected. That is, if D* is large then it is not valid to say there is no difference between the sample populations.

Critical regions are derived from

$$D^*_{\text{critical value}}$$

Neave and Worthington (1988) provide a table of critical values for D*, for values of 10%, 5% and 1%, but only for sample sizes less than 25.

Critical values for D [not D*] can be computed from the following:

$$D_{\text{critical}} = m \cdot 1{,}2238 \{n/n_A n_B\}^{\frac{1}{2}}$$

where m, for various confidence limits, is given by:

$\alpha_{2\text{ (two sided test)}}$	10%	5%	2%	1%
m	1,2238	1,3581	1,5174	1,6276

(The one sided test is not relevant to the simulation tests.)

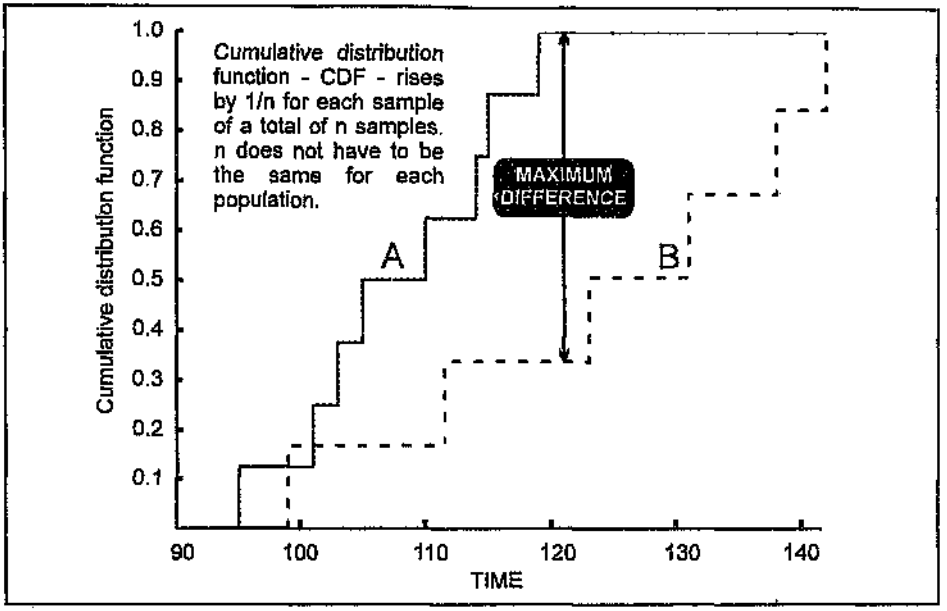


Fig. 4.6 Maximum difference shown between two hypothetical CDFs.

The Kolmogorov-Smirnov test is most appropriately used in instances where it is unclear what kind of differences to expect between the sample populations, or where expected differences do not fit into the usual categories of location (means or medians) or spread of the PDF. (Neave & Worthington, 1988).

The Kolmogorov-Smirnov test gives an estimate of how well the two sample PDFs match, and hence the reliability that can be placed on the other test of correlation performed. Its non-specificity towards types of differences in the PDFs make it much more suited to the types of data under consideration, than is the Mann-Whitney test, or its derivatives.

In common with most rank correlation type test, the Kolmogorov- Smirnov test also runs into problems with tied data. The first approach to this problem, suggested by Neave & Worthington (1988), is to simply ignore the ties. It has the disadvantage of *possibly* giving the statistic a smaller value than it should have, but the error is claimed to be small enough to have no substantial effect on the conclusion. The error is only introduced if the tied values occur in the vicinity of the maximum difference. The maximum difference refers to the largest occurring value $|Y_i - X_i|$. This may or may not coincide with the tied data points, depending on the nature of the simulation's failing. In the event of largely accurate simulations, the maximum difference will not occur near to tied values. A maximum difference coinciding with tied values occurs under conditions where a large outbreak occurred, yet failed to show up in the simulated results, or else where a non-existent outbreak was reflected by the simulation. The second approach is analogous to that suggested by Kendall - reordering the ties in all possible ways, and averaging the values of the statistics thus obtained. Once again, if the ties all occur at population values of zero, reordering will have no net effect on the test result.

A SIMPLE NON-STATISTICAL EVALUATION

A simple method of showing up good or poor correlations is to draw an X,Y scatter plot - as described earlier for testing linearity and monotonicity - which maps each point of the independent variable set to its corresponding point in the dependent variable set (Figure 4.7). Usually a line of varying degree of straightness results. No such line can be drawn when the vast majority of points in both the independent and dependent variable sets are zeros, as is the case with the simulation output and the associated count data. The goodness of fit line degenerates into a set of coincident points, which nonetheless do provide an indication of the quality of the simulation. If the points all lie at the origin, then they can be interpreted to lie on the line Y=X, where X and Y are the independent and dependent variables respectively. The number of points away from this line is then an indication of the poorness of the simulation.

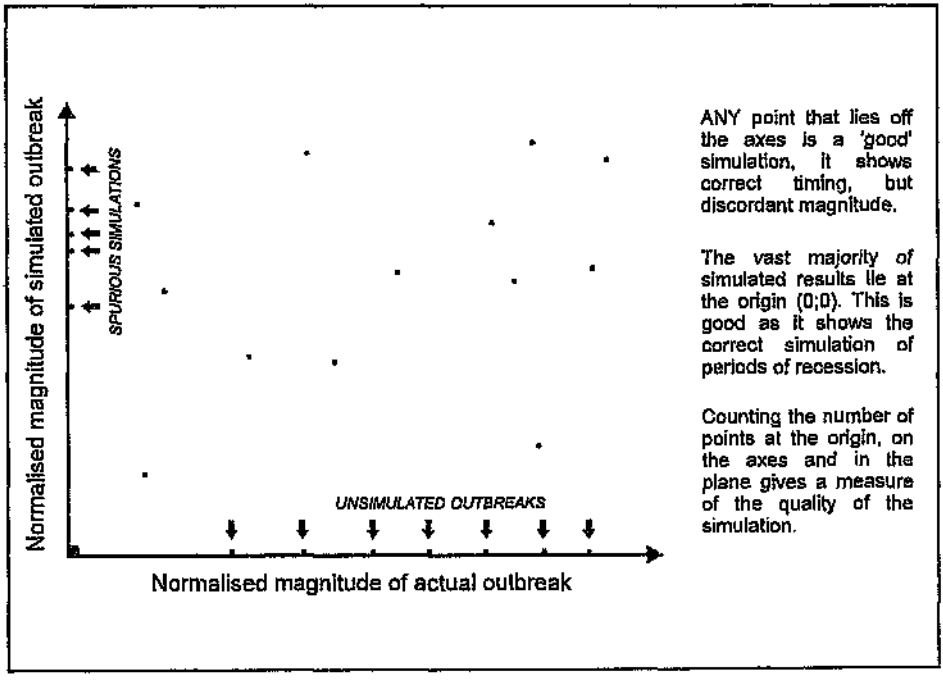


Fig. 4.7 Scatter plot of two random variables to illustrate graphically how a 'good' simulation can be determined by many points on the line Y=X, including those at the origin.

If the independent variable (considered here, with some licence, to be the actual count record) is represented by the horizontal axis, and the dependent variable, considered to be analogous to the simulated result, then the scatter plot may be interpreted as:

- points which lie at the origin show correct simulation operation, where inter-outbreak recessions were accurately predicted.
- points which lie below the 45° line (representing Y=X) reflect instances where an outbreak was observed to occur, yet the simulation failed to track it. In fact, only points which lie on the horizontal axis represent completely failed predictions. Any points within the lower triangle of the co-ordinate plane correspond to simulations of incorrect magnitude.
- points above the line Y=X reflect instances where spurious predictions were made by the simulation. As before, only points on the axis represent completely false outbreaks.

IN SUMMARY

The simulation model testing comprises three components: validation, verification and sensitivity testing. Test runs require weather data to drive the simulation, and actual locust count data to validate the simulation output.

Adequate weather data exist only for a few sites, and for a limited period (1986-1989). Within this interval, if weather data is sparse or missing, a technique of mean seasonal differences is used to synthesise data from applicable time periods. Special consideration is given to determining suitable initial conditions for simulation runs.

The Spearman and the Kendall rank correlation tests provide a quantitative measure of how well the simulation output maps to the actual count records. Neither of these tests take cognisance of the magnitudes of the outbreaks, but they do test whether large simulated outbreaks coincide with large actual outbreaks, and whether small simulated outbreaks coincide with small actual outbreaks. The simulation output data is unsuited to parametric tests. The Kolmogorov-Smirnov test is useful in determining whether the PDFs of the two samples (simulation output and actual count records) are similar. Similarity of the PDFs is a requirement if the rank correlation tests are to be meaningfully interpreted. Summaries of non-parametric statistical test results are provided in Appendix C.

In the following chapter, simulation results are presented alongside actual count data, a graphical qualitative analysis. The statistical test results are also shown for each set of simulation runs, providing a quantitative measure of the model's performance.

Chapter 5 COMMENTS ON TEST SIMULATION RUNS

5.1 OBSERVATIONS ON THE SIMULATION OUTPUT

The reader is referred to Appendix A (Simulation test results), which show the model output for the period 1986 - 1989, with actual swarms killed for comparison.

The timing of some of the simulated outbreaks is pleasingly accurate, even in the case of the non-migratory model. Note that when considering these results, it makes no difference which station is paired with which other station. Since no migration is allowed, all simulation results associated with a particular station will be the same, regardless of which station it was paired with during the actual test.

Notable results include simulated outbreaks at the following locations and times. Days indicated are approximate and are those shown on the graphs' horizontal axes, denoting days since March 1st, 1986.

SITE OF SIMULATION	DAY OF SALIENT OUTBREAK
Douglas	30, 60, 245
Graaff-Reinet	30, 60, 280, 315
Pofadder	30, 60
De Aar	245

The good tracking occurs predominantly in the early part of the simulation. This is not surprising since the initial conditions are those derived from actual outbreak numbers. If the locust population fluctuations are truly chaotic, as postulated, then the observed divergence from reality is to be expected.

The cause of concern is more the model's failure to simulate observed outbreaks, rather than the spurious simulations. For instance:

SITE OF SIMULATION	DAY OF FAILED SIMULATED OUTBREAK
Douglas	760, 790, 1005
Graaff-Reinet	395, 980, 1160, 1190, 1340, 1375
Pofadder	790, 950 - 1190
De Aar	760, 790, 980, 1005

The outbreaks, when they occur, are mainly triggered by rainfall events, but not necessarily exceptional rainfall. This highlights the value of having a genuine simulation model, rather than a predictor algorithm based on rainfall alone. When investigating the causes of the above errors, the primary focus should therefore fall on how the model interprets and applies the rainfall data, and under what conditions rainfall events become significant. These factors include quantities such as accumulated degree-days, the age of eggs, and preceding periods of wetness or dryness.

Rainfall charts of the four sites investigated are included in Appendix B.

Note that the statistical tests (Appendix C) show good correlation coefficients even in the case of the non-migratory model. This anomaly is explained further in

the section on statistical testing - section 5.2 - and also provided is an explanation of why the statistical tests are of limited value.

Inspection of the failed events listed above shows that they have all occurred towards the end of the simulation period, typically two years or longer after the starting date.

It seems unlikely that the failed events will be corrected by the introduction of migration emulation. All the non-migration simulations suffer the same drawback in that populations tend to die out within two years, leaving no adults or hoppers to potentially invade other areas. This extinction is not true of the egg reserves though, which can remain dormant for substantial periods - certainly longer than the dormancy shown by the simulation graphs. Thus the overriding problem seems to be one of eggs entering to deep a state of dormancy too early on in the simulation. It should be remembered, however, that the simulation model has by then executed several hundred time steps with no update from the actual count data. It is reasonable to expect quality of the tracking to diminish as time progresses.

Nonetheless, when migration is enabled, improvements in the prediction of late-simulation outbreaks results.

It has been suggested (Hanrahan 1998, pers. comm.) that the cause of the dormant egg problem outlined above might be in the ratios between gregarious and solitary females. Gregarious and solitary females tend to have different criteria when selecting laying sites (Price, 1988). The hatching of these eggs will thus be dependant on different conditions, a fact which the simulation does not take into account. Further, there is a distinct correlation between the phase of the adult and the type of eggs laid (Matheè, in Price, 1988). As mentioned earlier, the accuracy of density dependant computations (Wolda & Dennis, 1993) only becomes significantly useful after 20 to 30 generations - too long a span to be relevant in short simulations - which is why there is no consideration given to it in the model thus far - but which may become more influential as simulations much beyond 2 to 3 years are considered.

The Douglas ⇔ Pofadder simulation shows a curious result. Adult populations suddenly appear - when the migration is enabled - but there is no obvious source, since the other station of the pair exhibited no outbreaks at that time.

This can be ascribed to the fact that the model uses real (floating point) numbers to represent insect numbers. Small populations - smaller than one insect - can exist within the model, and do not show up on the graphs because of the log axes. (The logarithm of a fraction manifests itself as a number less than zero and so falls below the horizontal axis.) Very small "invisible" populations are migrating and thus introducing non-diapause eggs into favourable areas where there would otherwise be none.

In practice this is not a failing of the model, nor an error. It is just a manifestation of the initial population sizes being small (several tens to several hundred of individuals). In fact, the initial condition data used numerically represents swarm numbers, not individuals. Again, this is no reason for concern. The linearity of the

mathematical functions within each rule of the rulebase ensures proportional scaling no matter what the absolute sizes of populations. For example, if the initial populations were "increased" tenfold, the output would increase tenfold, yet still exhibit the same curve shapes and trends. Therefore, if the input data represents swarms, not individuals, then the output should be considered to represent the same units.

5.2 STATISTICAL ANALYSIS OF THE SIMULATION RESULTS

The statistical tests performed, that is, the Pearson product moment, the Kendall rank correlation and the Spearman rank correlation test as well as the Kolmogorov-Smirnov test, are presented for each simulation in Appendix C.

A common misconception concerning correlation coefficients is that any coefficient larger than +½ represents a good positive correlation, while any coefficient smaller than +½ implies a poor correlation. While it is true that numbers closer to "1" imply better correlations than those close to "0", the critical regions where the desired confidence intervals lie, are highly dependent on the sample size. Typically, for the simulation output, the sample size is 1402 or 1767, for simulations running to 1989, or 1990 respectively. No standard tables extend to this magnitude of sample size, but the appropriate quantiles for any available sample size are easily calculated from standard formulæ. Appropriate quantiles for the tests performed are shown below..

It can be seen from a glance at any of these tables that for usual levels of significance, α = 5%, α = 2,5%, α = 1%, and so on, the quantiles become exceedingly small as the sample size becomes large. For instance, at a level of significance α = 5%, the quantiles are as follows:

Spearman	R_S	$W_p = 0{,}0440 \mid_{n=1400}$	$W_p = 0{,}0399 \mid_{n=1700}$
Kendall	τ	$W_p = 0{,}0239 \mid_{n=1400}$	$W_p = 0{,}0266 \mid_{n=1700}$
Kolmogorov-Smirnov	D	$W_p = 0{,}0513 \mid_{n=1400}$	$W_p = 0{,}0466 \mid_{n=1700}$

(The Pearson product moment quantiles are not given here, because the joint probability function associated with $(X;Y)$ is unknown for the simulation data, and hence confidence intervals are meaningless)

Essentially what these quantiles mean is that even moderately "large" correlation coefficients imply good measures of correlation. It might be conjectured that the coefficients obtained from the test runs are exceptionally large in the light of these quantiles, but it is imperative to remember that, typically, 90% to 95% of all the sample points represent $(X;Y) = (0;0)$. It is therefore not surprising that the rank tests yield very strong measures of correlation, but also unfortunately makes these test results somewhat difficult to interpret within the more conventional understanding of correlation, and not particularly useful. It has been shown that the Pearson product test based on weighted covariance is unsuitable due to the unknown nature of the joint-probability function associated with the random variables. Under such conditions the rank based tests normally provide a suitable solution, but the assumption is that the data is more-or-less uniformly spread over the range covered, without too many zeros or ties. In the case of the simulation

output, and the actual count records, much of the sampled data consist of zeros. While this is entirely correct in representing the situation as perceived, it causes the rank tests to fail to give the desired indication of simulation accuracy. It might be suggested that the zeros be deleted from both records and the focus placed on those instances where locust numbers rose substantially. These events occur infrequently during the test period, and the sample size would tend to make the tests less than satisfactory.

Intuitively, the large number of zeros indicates that - for the most part - the simulation is operating correctly. Much of the simulation period involves records that show no outbreaks. All the non-existent outbreaks correlate well in an abstract sense with the simulation's null output.

The rank correlation coefficients might seem in the light of this description to be giving a good representation of the quality of the simulation. But, it is easy to imagine that a pseudo-simulation, the output of which remained continuously at zero, would not give rise to a much different result in this respect.

A further complication is that the rank correlation tests behave "well" when there are not many ties present in the data. In order to make the statistical analysis engine built into the simulation operate at an acceptable speed, ties (in the case of the Spearman's coefficient) are only compensated for when they occur at zero. This accounts for all the ties present in the actual count data, and the vast majority of the ties in the simulation results. Regions of non-zero tied data points are easily identified in the graphs as flat-topped. Conover (1987) suggests a different formula to be used for calculating Spearman's rank when a *large* number of ties are present, and this result is included in the statistics summary.

It can be seen that the corrected Spearman's coefficient , in some cases, falls as low as the p=0.7 quantile, and so indicates a much poorer tendency towards correlation. But in the light of the number of ties occurring, this number should not be given too much credence.

The Kendall coefficient of rank correlation is easier to compute in terms of program complexity, and does not require such extensive computer processing of the data to account for ties. For this reason, the computed coefficient is more accurate, although the Spearman's rank coefficient cannot be compared directly to the Kendall, coefficient since their critical regions are completely different.

Pearson's product moment is calculated, using the standard formula, but as discussed earlier, since the joint probability function of the two data sets is unknown, the test statistic has no meaningful interpretation. Its value is included only to highlight that numerically it can give misleading results.

Thus the simulation generates correct results for most of the test periods, but the numerically high correlation results may cause the output to be given more credence that it merits.

5.3 NON-STATISTICAL EVALUATION

As described in the previous chapter, one graphical method of showing up good or poor correlations is to draw an X,Y scatter plot. To re-iterate:

- points which lie at the origin show correct simulation operation, where inter-outbreak recessions were accurately predicted.
- points which lie below the 45° line (representing Y=X) reflect instances where an outbreak was observed to occur, yet the simulation failed to track it. In fact, only points which lie on the horizontal axis represent completely failed predictions. Any points within the lower triangle of the co-ordinate plane correspond to simulations of incorrect magnitude. Points *on* the x-axis represent completely failed simulations.
- points above the line Y=X reflect instances where spurious predictions were made by the simulation. As before, only points *on* the y-axis represent completely false simulations.

A simple analysis and evaluation technique that avoids all the pitfalls of the statistical computations and their requirements for meaningful interpretation is to generate data for such a scatter plot, but the plot itself is not necessarily required. A 'quality index' for the simulation can then be computed by counting *all* the data points that lie on the line Y=X (including those at the origin, Y=X=0), and expressing this total as a proportion of the total number of points (which will be the number of days for which the simulation was run).

In an analogous manner, an 'index of spurious prediction' can be computed by counting up all the points that lie on the Y axis, and an 'index of failed predictions' computed by counting points that lie on the X axis.

The obvious caveat in this technique is that any timing errors - even if as small as one day - in predicted simulations cause a large drop in the quality index. This is particularly important to note since the actual count records (more accurately, the control records) are monthly cumulative totals. Modifying the simulation software to provide its output as monthly cumulative totals would provide a satisfactory solution to this problem. From a software point of view this is not difficult, but it does require an accurate calendar to be built into the model so that account can be taken of 30- and 31 day months, February, and leap years. A simple 30 day cumulative total technique would go out of synchronisation with the actual count/control data. It would, however, be better to obtain count data with specific dates. Then the association between specific rainfall events and outbreaks could be more closely examined. The simulation output has been retained as a daily population plot so that this very phenomenon can be investigated, at least during simulations, if not in the case of count records.

Notwithstanding the fact that this is the most simple of tests, it is the one that is least disrupted by the unusual type of data under question, and is regarded as the test which provides the best indication of the model's performance and improvement in performance.

Even when applying the most stringent test condition of prediction accuracy to a single day, the scatter plot test showed the notable improvements (Table 5.1).

Table 5.1 Scatter plot analysis showing improvement in number of off-axis XY pairs when migration is enabled.

no migration	—	with migration	no migration	—	with migration
	Douglas			Graaff-Reinet	
4	•	9	3	•	7
	Douglas			Pofadder	
4	•	5	5	•	13
	Douglas			De Aar	
4	•	6	2	•	10
	Graaff-Reinet			Pofadder	
3	•	6	5	•	4
	Graaff-Reinet			De Aar	
3	•	6	2	•	9
	Pofadder			De Aar	
5	•	3	2	•	8

Note that these numbers refer to the number of days during the simulation period (1986-1989) on which the model provided good temporal accuracy. The tabulated values refe[illegible]e number of off-axis XY data points, in other words, those points that do not [illegible]ent spurious or failed simulated outbreaks, but rather simulated outbreaks that coincided time-wise, but not necessarily magnitude-wise with actual count records. It must be remembered that the actual count data is a monthly cumulative total, and hence the term 'accuracy' is used with some license. What is important to note is the increase in number of accurate simulation events when the migration engine is enabled.

These data show an improvement in the simulation performance in all cases but one (Pofadder). The small numbers might initially appear disappointing, but since accurate timing is of paramount importance for this test to work, the inclusion of some form of "temporal windowing" (for example, using simulated monthly totals) is anticipated to dramatically increase these values.

In biogeographic studies, commonly encountered measures of similarity between biotas - real or simulated - are, *inter alia*, the indices of Jaccard, Dice and Simpson. Brown & Lomolino (1998) define, where
C = taxa present in both units,
N_1 = total taxa present in first unit,
N_2 = total taxa present in second unit,

Jaccard coefficient: $I_j = C \div (N_1 + N_2 - C)$
Simpson coefficient: $s_i = C \div N_1$
Correlation ratio: $r_c = C^2 \div (N_1 N_2)$
Dice ratio: $r_D = 2C \div (N_1 + N_2)$

These are all used to quantify the similarity between pairs of biotas, and are based on simple presence/absence data. Several other similar coefficients are given, all based on various algebraic expressions involving C, N_1 and N_2, which are conveniently visualised by means of a Venn diagram, as in Figure 5.1.

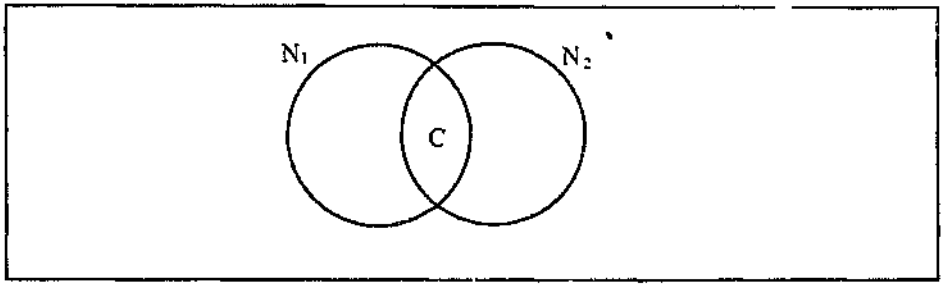


Fig. 5.1 Venn diagram showing variable definitions for the Jaccard, Simpson, Dice and correlation coefficient indices.

In essence what these tests *all* do, is express the shared (common) events (populations present) as a proportion of some expression of the total events possible.

The failings of all of these is that they cannot address the question of size-zero populations, (a problem of many otherwise useful tests, and a notable point being that any test must give the model a thumbs-up when it correctly simulates no outbreaks), **unless** one considers these no-size populations (no outbreaks occurring) to be some kind of special 'population' where we can't numerically state the population size, but we can say it is the same in both regions (i.e., actual counts and simulated population). In that case, the zero size populations **do** feature in the Jaccard coefficient (and others), and then numerically the Jaccard coefficient gives the same result as the suggested method of using a scatter plot.

It has been suggested (Piper, 1998, pers. comm.) that a χ^2 test could be used to evaluate the magnitude accuracy of the simulations. This commonly encountered test measures whether the outcomes of a number of observed events are due to chance or due to an underlying cause. The χ^2 test does not work on continuous variables, but on expected frequency of events. In this sense, one would have to calculate the frequency of an outbreak of a particular size, and do so for every possible magnitude of outbreak. Then the chance of getting an outbreak of that size category is used in the test. In that way, the timing of outbreaks falls away entirely, which has the potential for making the statistic completely misleading.

Suppose the simulation - for whatever reason - generates outbreaks that are coincidentally of the right size, but they appear at the wrong times: the χ^2 test will yield a high quality index, although equivalent magnitudes do not coincide timewise. Logically, one cannot evaluate the outbreak sizes on their own without considering the timing of the outbreaks, unless one does it on a day-by-day basis In this sense the author originally considered computing the "least squares error" as a quality indicator for outbreak magnitude. This technique compares outbreaks to simulations on a day-by-day basis over the whole simulation period, and totals

the errors found on each day. The reason for discarding that test was that the result is an arbitrary number, with no scale of reference, so it is impossible to say that a simulation with a least squares error of say 1000 is better than a simulation with a least squares error of 900; the values depend on the actual magnitudes of outbreaks. The only useful application is when comparing multiple simulations on the **same test data**, where the user has changed some model parameter. Then the reduced error as determined by the least squares shows up a better or worse model. With hindsight, the least squares error computation could be modified to reflect a proportional error, rather than an absolute error, making different simulations directly comparable.

5.4 SENSITIVITY TESTS

These tests involve choosing a parameter on which migration computations are based, and varying that parameter about the expected correct value. For each altered value of the parameter, the simulation is re-run (all other conditions remain unchanged). The population fluctuations are then compared for the whole range of values of the parameter under consideration. The "importance" of a particular parameter can thus be gauged by the contribution changes in that parameter make to the model output.

There are a very large number of possible sensitivity tests that could potentially be performed, especially when considering all the parameters that are available for user-adjustment within the locust lifecycle model. However, since the focal point of this report is the impact of migration effects on the simulation, only the sensitivities with respect to those parameters are shown.

In doing standard tests, it was discovered that the model in all forms (migratory and non-migratory) is exceptionally sensitive to initial conditions. Small changes in the relative initial populations can cause large changes in the magnitudes of outbreaks one or two years later. Such properties are typical of chaotic systems.

SENSITIVITY TEST CONDITIONS

Sensitivity tests are conducted by varying the parameter of interest about the nominal value, i.e., that value which is assumed theoretically to be most appropriate, shown in italics in the formulae below. The nominal value is chosen according to the background and modelling theory outlined in Chapters 2 and 3. The range of values over which the parameter is varied is chosen to give a span which might reasonably show fluctuation is population magnitudes, or in cases of time and temperature, to be ranges that might reasonably be encountered in reality.

Tests with regard to wind speed and direction computations

Sensitivity was tested with espect to the values of β an *b*, in the following probability equations:

1. for the wind magnitude computations

$$P = a.e^{-h(s-s')^2} \quad a = 1,\ h = \ln_e(\beta)/s'^2$$

2. for the wind heading computations

$$P = e^{-k(\delta - 180°)^2} \quad k = (\ln_e b)/180^2$$

β and *b* are independently varied in the range [70 80 90 *100* 110 120 130]

Tests with regard to flight threshold temperature

Varying the threshold temperature for flight about the 'nominal' or generally accepted value of 20°C

$T_{threshold}$ is varied with the range [9 12 15 18 *20* 21 24 27] °C

Tests with regard to flight time limiting

The process that determines flight time is potentially subject to errors in that if the daily temperature never drops below the flight threshold, then migrations are not terminated (flights may endure for 24 hours). To avoid this problem, a limiting case has been introduced, which is simply the number of hours after which a migration event is terminated no matter what the temperature. This is intended to model the exhaustion of insect lipid reserves, although it does not take biological factors into account.

This value is nominally set to 20hours. If the value exceeds 24, then it is conceivable that migrations may then endure for an unlimited period - if the temperature is conducive to flight - since the following day might exhibit sufficient temperatures for flight, with no forced cessation between days. This phenomenon is evident in a levelling off of the sensitivity curves beyond the 24 hour limit.

Fly_time_{max} is varied in the range [11 14 17 *20* 23 26 29] hours.

STATISTICAL ANALYSES ON SENSITIVITY TESTS

It is feasible to perform correlation coefficient tests on these simulated populations. For example, each altered value for beta (β) requires a simulation run. Thus each of these simulation outputs could be correlated with the actual count record to achieve a set of correlation coefficients relating to the set of values for β. However, in the light of what will be discussed concerning the correlation coefficients, it was felt that these numbers are not particularly useful, and a better feel for whether the simulation is improving or worsening can be obtained by inspection of the population/time graphs. A rigorous analysis of sensitivity to

migration parameters should ideally only be performed after the sensitivity to lifecycle parameters is known.

PRESENTATION OF SENSITIVITY ANALYSES

The sensitivity tests are only performed using one pair of weather stations - those which gave the best subjective output during the validation tests, Douglas and Pofadder. The sensitivity of the model to migration parameters is not a function of any of the population magnitudes, so it makes little difference which simulations are used for the sensitivity test. The only condition that must be satisfied is that some migration must occur during the simulation. The Douglas ⇔ Pofadder simulation has been selected as it illustrates a noticeable variation when the migration engine is switched on.

The sensitivity tests are presented as a suite of curves: population magnitude as a function of time. Initially these appear quite complicated: the simulation is run for each value of the parameter under scrutiny. Predictably then, the population fluctuations are different each time the simulation is run. The population/time plots for each parameter value are presented side-by-side on the same population/time/ parameter axes, thus forming a 3-D surface rather than a 2-D line.

The function of the population magnitude is plotted on the y-axis (vertical axis). The time progression of the simulation (number of days) is represented by the horizontal x-axis, and a third component - the parameter under investigation - expands the plot into a 3-dimensional surface, obliquely towards the viewer.

In this way it can be seen that as the parameter of interest changes, it may have little effect on population sizes, or it may cause population sizes to be different only at certain times, or it may cause the population sizes to be different over the whole simulation period.

The population magnitude component has been modified in its representation. The vertical axis represents the *deviation* in population size on each particular day, as compared to the corresponding simulated size before the parameter under consideration was altered. For example, suppose the total outbreak magnitude was stored as a matrix variable called *Population*, and this was simulated using a flight threshold temperature of 18°C. It is convenient to note the population magnitude as $Population_{18}$. Now suppose the parameter, flight threshold temperature, is modified, yielding a different simulated population, say, $Population_{20}$, and so on.

The sensitivity plot is generated using the functions of the form:
$Population_{18} - Population_{20}$,
$Population_{18} - Population_{22}$,
$Population_{18} - Population_{24}$ and so on.

The advantage of this is twofold:

1 it makes the 3-D plots easier to see, the curves being flatter (the range of the population axis is small) and less likely to be obscured by those in the foreground,

2 the scaling is more conducive to highlighting small fluctuations, which would otherwise be "lost" owing to the largest values having to be accommodated on the axis.

Notice that in this formula, the adjusted population forms the subtrahend (that which is taken away). The simulated population for the unchanged parameter (nominal value) corresponds to a straight line (no fluctuations), and the curves show differences from the nominal, in response to the variation of the parameter under consideration.

Flatter surfaces show relative insensitivity to the parameter. Surfaces with pronounced peaks and troughs indicate a marked sensitivity, since the change in the model output is large in response to changes in the parameter.

Presented overleaf are the sensitivity test result plots for Douglas and Pofadder for the period March 1986 through December 1989. Sensitivity test were performed on the Douglas⇔Pofadder simulation in all cases, thus the top plot represents populations for Douglas, and the bottom plot represents populations for Pofadder in all four pairs of graphs.

Note that some of the 3-D plots are viewed from "behind", to prevent the plane from obscuring the negative peaks. Thus the day axis increments from right to left.

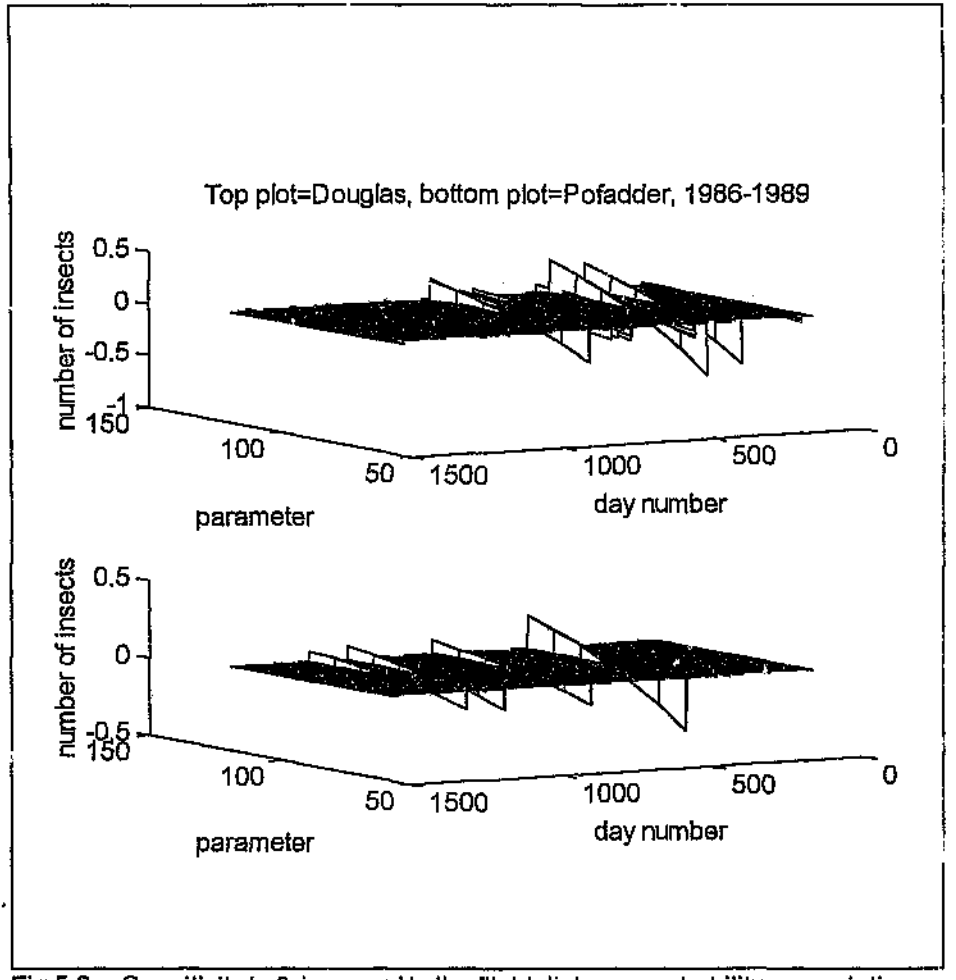


Fig 5.2 Sensitivity to β, in regard to the flight distance probability computation

$$P = \exp\left\{\frac{-\log\beta}{s'^2}\cdot(x-s')^2\right\}$$

with parameter β assuming values [70 80 90 *100* 110 120 130], nominal value shown in Italics.

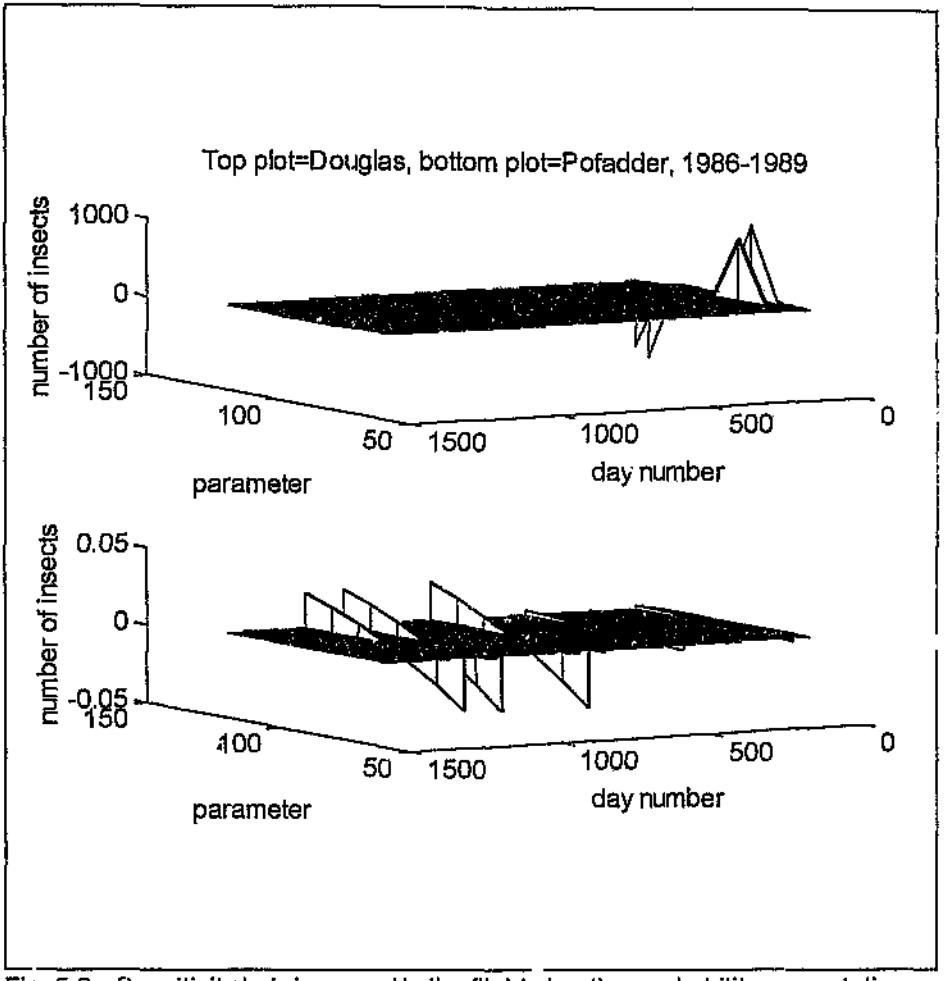


Fig. 5.3 Sensitivity to *b*, in regard to the flight direction probability computation

$$P = \exp\left\{\frac{-\log b}{180^{\circ 2}} \cdot (\delta - 180^{\circ})^2\right\}$$

with parameter *b* assuming values [70 80 90 *100* 110 12[illegible] 130], nominal value shown in italics.

INTERPRETATION OF TESTS WITH REGARD TO WIND SPEED AND DIRECTION COMPUTATIONS

The model is insensitive to changes in the value of β (Figure 5.2). That is, the spread of the wind direction curve does not dramatically affect the magnitudes of simulated outbreaks. Population changes of less than one are noted for changes in β of ±40. (β is a dimensionless constant). This insensitivity can be ascribed to the low (one in sixteen) resolution of the wind bearing readings.

The model is very much more sensitive to the parameter *b* (Figure 5.3), which governs the spread of the flight-distance probability as related to interstation distance. In this case the computations include time of flight information too, which introduces a much finer division of measure to the parameter, and hence a greater sensitivity. Notice that this does not apply at both stations though. Population fluctuations at Pofadder remain insensitive to these parameter variations, and also note that the sensitivity at the Douglas station is confined to events early in the simulation.

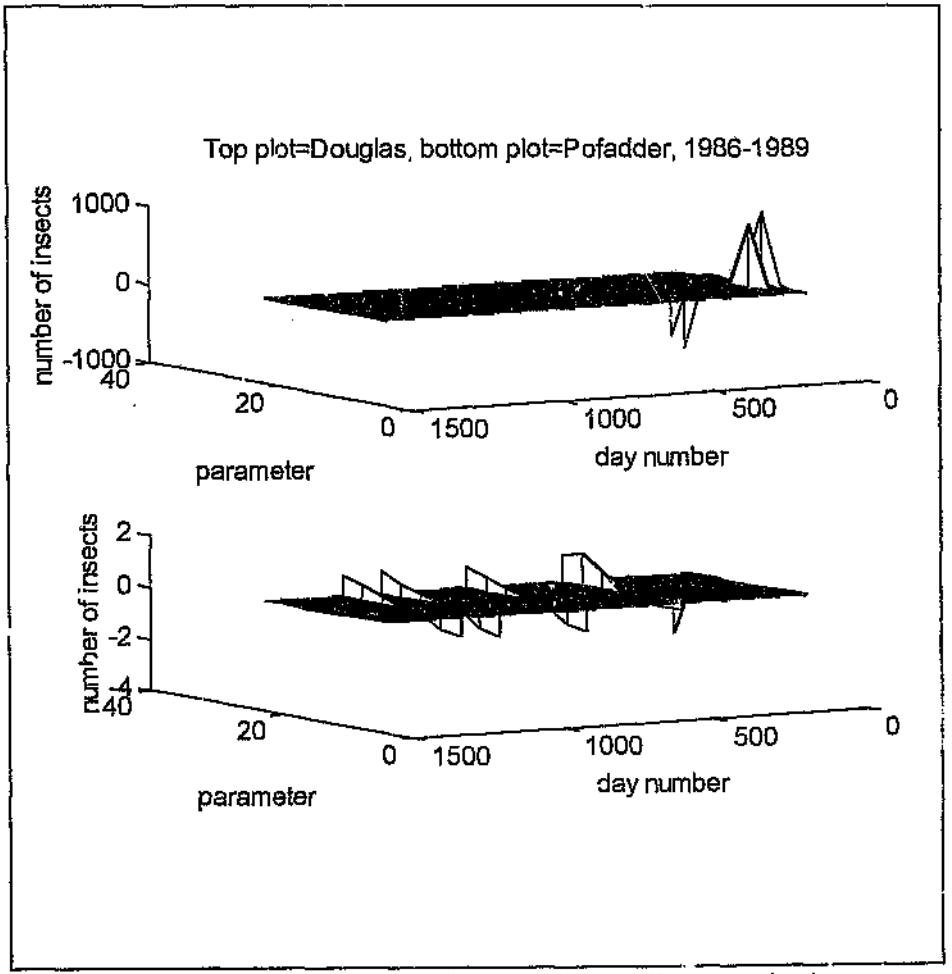


Fig. 5.4 Sensitivity to flight threshold temperature, with parameter *temp* $_{threshold}$ assuming values [9 12 15 18 *20* 21 24 27] °C, nominal value shown in italics.

INTERPRETATION OF TESTS WITH REGARD TO FLIGHT THRESHOLD TEMPERATURE

The reader is referred to Figure 5.4.

Similar trends to the sensitivity to *b* apply. At Douglas, sensitivity is marked only early in the simulation period. Sensitivity at Pofadder is much lower than at Douglas.

An interesting feature is the levelling off of the sensitivity curves as the parameter deviates far from 100 (either higher or lower).

At Douglas, the curve shows unexpected behaviour in that when this parameter is lowered substantially below the nominal of 18°C, the sensitivity drops.

Interestingly, this behaviour is seen also in the case of flight time limiting and *b* sensitivities in the case of Douglas simulations.

This anomaly is certainly grounds for further investigation. There is the possibility that there is a problem in the Matlab 3-D surface, since this software procedure failed to operate correctly on other occasions.

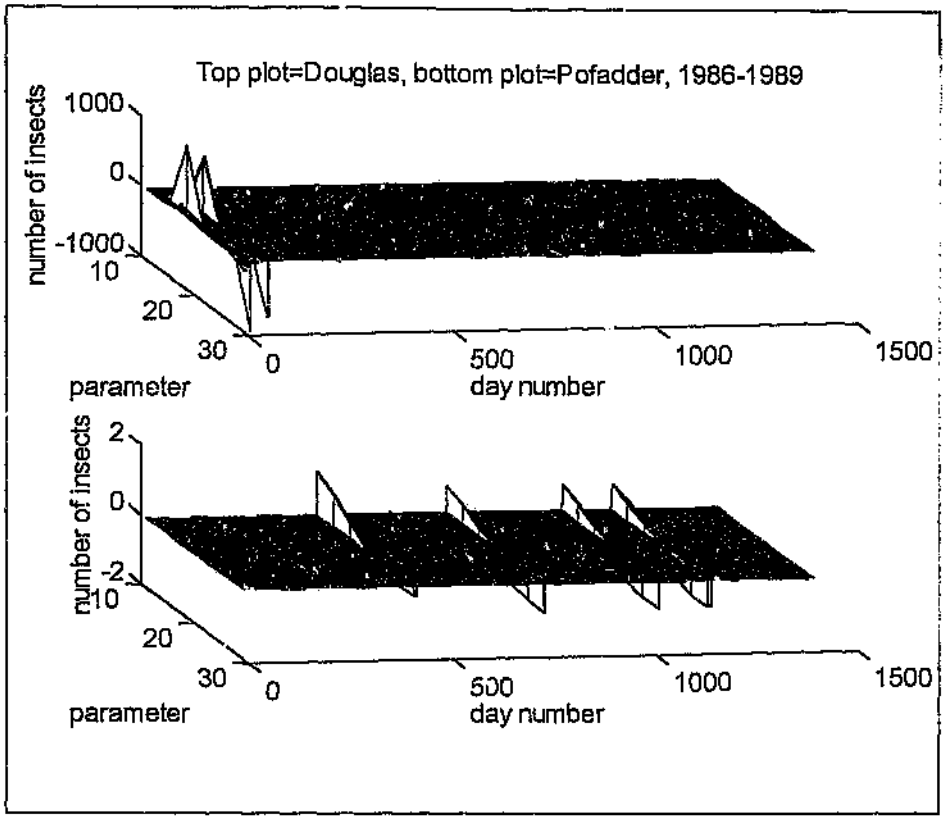


Fig. 5.5 Sensitivity to flight time limiting value. with parameter *fly-time* $_{maximum}$ assuming values [11 14 17 *20* 23 26 29] hours, nominal value shown in italics.

INTERPRETATION OF TESTS WITH REGARD TO FLIGHT TIME LIMITING

Once again, high sensitivity is evident at the beginning of the Douglas simulation, and noticeably so in the early part of the simulation, as shown in Figure 5.5.

Understandably, the sensitivity curve levels off as the parameter is raised much beyond 24h, since the subsequent day's high enough temperatures would have initiated flights rather than flights being "carried over" from those initiated on the previous day.

It is evident that the sensitivity to these parameters - as anticipated - is closely linked to the nature of the prevailing winds, and not simply the way the model processes that data. A better understanding could be gained by supplying the model with fictitious, carefully designed test wind patterns.

In all cases, the trend is for the sensitivity to drop as the simulation progresses. This is in line with the observed phenomenon of population extinction as time progresses. As flying adults are depleted from the simulated populations, so the dependence on factors related to migrations becomes progressively less relevant.

Chapter 6 COMMENTS AND CONCLUSIONS

6.1 GENERAL COMMENTS

The simulation model has been improved considerably since its first adaption to run on a personal computer. Notable advancements are:

- Several errors in the application of the rule-base have been identified and corrected.
- The total memory requirement of the simulation has been reduced from approximately 50MB to under 5kB (excluding overheads required by the interpreter). This is a considerable achievement in progressing towards a true multi-point grid model which can simulate several geographic locations simultaneously.
- The execution time has been reduced from tens-of-hours to under five minutes (single site simulation, four year data span). This allows users the chance to experiment with various conditions and parameters and achieve visible results within a reasonable time.
- The two site model presented here contains all the necessary elements for expansion to a 3 or 4 or more site simulation. It is expected that the model can be expanded considerably before a larger, faster computer becomes a necessity.
- The soil temperature model developed as an essential side-project. This model will find useful application in a number of other soil-dependant phenomena, both in the biological field and outside.
- The statistics tests are built into the simulation. In principle there is no requirement for additional analysis tools, although at present the statistics calculation unit requires refinement in terms of assumptions and approximations which it makes.
- The Windows graphical user interface (GUI) is now an integral part of the simulation. This includes modules to generate output plots. Most people familiar with Windows applications should find that experimenting with the model requires an acceptably brief learning period.
- A full-feature user manual and technical reference has been compiled, providing for both novice users and the more advanced users who wish to adapt to model to other applications, or improve its performance (Appendix E).

6.2 NEGATIVE ASPECTS OF THE SIMULATION ANALYSIS

The primary problem encountered in performing the simulation test runs was optimising the locust lifecycle. As already discussed, there are many difficulties that arise from models becoming too intricate, and this has been confirmed to be so with the locust simulation model. The lifecycle model has become large with regard to the number of parameters which are considered when applying the rulebase. The solution to this difficulty requires a large number of test runs, once good climate data has been assimilated. By varying the lifecycle parameters between successive test runs it will be seen which parameters are responsible for which changes in the output. In summary, a thorough sensitivity analysis is required in terms of the lifecycle parameters.

The same software analysis module developed for the migration parameter sensitivity testing can be used for analysis of any of the lifecycle parameters.

Although these parameters are thought to be accurate in the light of research done by Price (1988), and Mattheè (1951), the errors in the simulated outputs show a definite shortcoming in the model. It must also be noted that much of the research on the Brown Locust and its egg physiology has been carried out under controlled laboratory conditions, which may not be applicable in the field.

In addition to this, the test input data is of dubious accuracy. The weather data is assumed to be accurate for periods where it exists, and where simulation failures coincide with periods of synthesised weather it is easy to attribute the failures to errors in that data. Unfortunately, during intervals described by accurate and contiguous weather data, there is still no guarantee on the accuracy or completeness of the actual locust count data. The locust count data is undoubtedly in error to some degree. The swarm reports are based on chance sightings and control attempts, and not followed by accurate surveys. Further, the magnitudes of the reported swarms killed are not based on any accurate or well established method of determination. The possibility of their being unrecorded swarms is imagined to be high, and the magnitude of many reported swarms is likely to be larger than the records indicate.

Manipulating the model parameters to fit this observed data will result in an unrealistic model. Alternatively there may be no correct solution to the problem of optimising the parameters, since a fictitious phenomenon is being "observed" (i.e., the count records give an inaccurate picture).

If the actual count data is taken to be reasonably accurate, then a salient characteristic of the model is that it tends to produce false alarms more often than it fails to simulate actual outbreaks.

The simulated output plots show immediately that locust populations tend to die too readily. In other words, the simulations track actual events reasonably for a variable period of one to two years, but then tend to simulate smaller and smaller outbreaks. The correction of this lies in tuning the parameters which trigger egg hatching, since the interpretation of this is that eggs enter too deep a state of dormancy (quiescence) to hatch with typical rainfalls.

Before any definite causes for the simulation errors can be identified, additional tests must be performed under conditions that meet the following criteria:

- Accurate soil temperature data needs to be obtained.

  Although it has been accepted that none of the sources of weather data available during the duration of this project (SAWB, ISCW, CCWR) can supply soil temperatures, it may be possible to find a source of such records. If not, then the technique used to emulate soil temperatures should be analysed more fully on its own, and not within the conte:' ~f the locust simulation model where other issues can cloud - or completely mask - shortcomings or triumphs of this method.

- Accurate count data needs to be obtained.

  What has consistently been referred to as "count data" is in reality a count of locust swarms killed, or hopper bands killed. These records therefore do not accurately reflect locust numbers, since no measure is taken of the size of the swarm or band. These numbers therefore represent a component of the population removed, and may prove more useful in future as an adult- or hopper mortality component to the model.

  Additionally, only those swarms or bands which are identified and killed are counted. There certainly exist a much larger number of populations which do not get seen.

- Egg bed data needs to be available.

  The combined hopper band and locust swarm data generated by the model represents only a fraction of the information that the simulation generates. Recall that all the egg stages are available at the simulation output too, but if this cannot be compared to real data, it is not useful in evaluating model accuracy.

  Further, the egg beds constitute the lion's share of an accurate simulation's initial conditions. All the simulations presented here have been performed with no egg bed reserves. It is quite possible that accurate initial egg numbers will generate better simulations. These initial conditions were omitted at present because it was felt that introducing artificial or invented egg numbers would be less helpful than assigning all these records to zero. Inaccurate numbers comprise a form of noise, or a perturbation input to the model, which is understandably undesirable.

- More sites need to be available, in closer proximity.

  The four sites used for test simulations are a great distance from one another (see figure 4.6), placing a great dependence on the accuracy of the wind data and associated computations. It would be desirable if simulations could be concentrated in small areas, where migration distances are short and the movements could be tracked more closely.

6.3 POSITIVE ASPECTS OF THE SIMULATION ANALYSIS

Some of the simulated outbreaks are simulated to a surprising degree of accuracy with regard to timing. Recall that the actual count data is a monthly cumulative total, and thus the model can never be tested to a temporal accuracy of less than 30 days. However, if the run is performed over four or five years , this resolution corresponds to an error less that 2% in either case. This should be kept in mind when zooming in on a short period simulation, since the error becomes proportionately larger as the examined time span decreases.

The observed accuracy of the timing - when it occurs - can be attributed to the initiation of outbreaks by rainfall events. When comparing simulation plots to rainfall plots, the coincidence of these two is striking. Hence the timing of an outbreak in response to rain is not, in fact, that surprising. According to Price (1988), hatching occurs approximately two weeks after a rainfall event - other conditions being favourable - but this delay is masked by the 30 day resolution limit imposed by the count data.

The population/time graphs do show in some instances a striking improvement when migration is allowed. Outbreaks that were conspicuously absent in the non-migratory model appeared - although with errors in timing - once migration was allowed.

The scatter-plot technique shows a quantitative improvement in the accuracy when migration is enabled. The smallness of the change, and the method's simplicity should not mask its value, especially if more accurate count data becomes available.

6.4 KNOWN FLAWS IN THE SIMULATION MODEL

Features of the locust lifecycle and migration that have been outlined in the cited references, and which are not supported by the model at the time of writing are:

- The flight threshold temperature used is compared to that measured at ground level, when evaluating the possibility of flight. Although this threshold is almost universally considered to be approximately 20°C (Botha & Jansen, 1969; Clarke, 1971), no account has been taken of the change in temperature with altitude. This *environmental lapse rate* is of the order of 0,6°C to 1°C per 100m, depending on the vertical mixing of the air and the humidity. Under any circumstances, since only near-surface flight activity is under consideration, there is no necessity to take the temperature profile into account. High altitude flight has been ignored.
- The model does not support natural changes in the locust populations.

  The ratio of solitarious phase to gregarious phase insects remains fixed throughout the simulation period. Converting this fixed parameter to a variable based on density dependence might help correct the problem of eggs prematurely entering a state of deep diapause and thence stifling the continuance of swarming populations. A potential problem is the lack of

information concerning those field conditions which stimulate changes in phase. These events are only envisaged to have an effect on long period simulations.

- Poor resolution of the wind speed data. The wind speed data is only recorded to a resolution of 5 knots. Light winds may occur but not be reflected in the data set, which means that low windspeeds (under 5 knots) imply no migration. Initially a migration probability of 1% was allowed in the absence of ANY wind records (missing data), then there was still some migration simulated north to south, since the default wind heading is 0° (a non existent value is interpreted as a zero by the data processing software). It is not desirable to suppress such events totally since other sites may have high winds in the right direction, and then a valuable migration event might be inadvertently suppressed. However, in the cases of missing data, the migration of populations to false destinations was felt to be more of a problem than the suppression of such a migration event.

- No migration of maturing adults occurs in the current model. This is biologically an important time for migration to occur. However, this error is not envisaged to alter the output dramatically. All instars progress to maturing adults, and experience a fractional depletion to represent mortality and predation. Thus the quantity of maturing adults is always a linear proportion of the number of hoppers present, offset in time by the accumulation of the necessary degree days. These populations of maturing adults precede the populations of mature adults also by the accumulation of the necessary degree days. Thus all that can be expected to change is the magnitude of migratory populations, and the timing of migration-induced outbreaks, but this will fall below the lower resolution limit imposed by the actual count records, and therefore its significance will be masked.
  Adding an extra migration pathway for maturing adults is not an involved procedure and might become necessary if more accurate test data becomes available.

- The wind speed magnitude high enough to completely depopulate a region is unknown. The assumption is made that increasing wind speeds - up to a cut-off threshold - will increase the magnitude of an emigrating swarm. As discussed previously, there is no accurate information that allows the model to determine exactly how much of a population should be removed for any particular wind speed. Correction of this information is likely to adjust the magnitudes of migration-induced outbreaks, but be unlikely to show up any "new" outbreaks, or inhibit any that are currently being simulated.

- The simulation operates only at two geographic locations. This is the first phase in the creation of a multi-point model that supports simulations on a network of interconnected nodes. As such, this is not seen as a flaw, but rather an expected limitation at this level of development. Nonetheless the lack of source sites, for immigrating individuals could contribute to the early extinction of populations observed during long simulations.

- The Spearman's rank correlation coefficient calculation makes the assumption that tied data values only occur at (0;0). While this is true in most instances investigated, it is a source of error in this statistic when ties occur at non-zero values. The reason for this assumption is to make the execution of the computation acceptably fast.
- None of the rank correlation tests are ideal tests of the simulation's success. Other commonly encountered statistical tests are unfortunately even less appropriate to the kind of data encountered in the simulations. At present it is felt that if a number has to be assigned to the quality of a simulation, then the best indication of how well the simulation performs can be gauged by counting the number of correct simulations, expressed as a ratio of spurious predictions. The term "correct simulations" is open to subjective interpretation, since the window allowed for the correct timing of an event is a personal preference, as is the magnitude error allowed.
- The simulation output needs to be converted to monthly totals to be more accurately compared with the count data. Alternatively, and more useful, if the count data could be presented on a daily basis and not as monthly cumulative totals. This would allow more accurate pinpointing of simulation errors, since these could be analysed in the light of prevailing weather conditions on that day.

6.5 OVERALL CONCLUSION

The initial development objective of the project was:

1. Is it possible to determine migration movements from wind speed and wind direction? Can this be done accurately enough in order to achieve the objective outlined in "2" below?
2. Will the accuracy of outbreak predictions improve significantly if the simulation model is expanded to take migration movements - based on wind phenomena - into account?
3. Will taking migratory movements into consideration create suitable initial conditions for the next season? This differs subtly from "2" in that it will take into consideration migratory effects, such as population relocation, and dispersal of eggs over many seasons. The question addressed in "2" is directed more towards finding out if population redistribution has an immediate effect.

The following answers are tendered:

1. Yes. In some instances the wind speed and direction is sufficient information to correct errors in the simulation that arise when migration is not taken into account. However, these wind parameters alone do not satisfactorily account for the errors observed.

2 No. The outbreak predictions (simulations) improve, but statistically there is no significance in the observed change. A better method of evaluating the simulation is required.

3 Yes and no. The simulation model generates relatively good initial conditions for one subsequent season whether migration is allowed or not. This can be seen by the quality of the tracking over the first 600, or so, days. Beyond this period, the simulation fails to track well, there being virtually no hatching in subsequent seasons (beyond approximately 700 days) whether migration occurs or not.
In some instances, the migration simulation provided very interesting and pleasing results (Douglas ⇔ Pofadder), but these improvements are not universal to all sites and events.

It is theorised that large errors in the lifecycle parameters are masking the more subtle effects of the overlying migration model.

6.6 RECOMMENDATIONS

During the writing of this report, and based on evidence inferred from the model test phase, the following amendments are suggested for improving performance of the simulation model:

- The biological component of the lifecycle model has not been tested to a satisfactory degree. The effect of varying parameters involved in driving the lifecycle model should be considered before any major adjustments or changes are made to the migration component. This is not to say that the migration component is not without flaws, but if errors exist in the underlying substructure of the biology, they can not be corrected by processes that occur in the migration unit.
- The model is too demanding on climatic data. The need to provide soil temperatures makes the number of useable weather stations available inconveniently low.

  It is not advisable to substitute air temperature for soil temperature without some corrective procedure. Unfortunately, accurate corrective processes require other data - such as soil moisture content, soil type, cloud cover (Thunholm 1990, Mathèe 1961), which is more difficult to obtain than is the soil temperature.

  It has been shown that the simple mapping function of mean seasonal differences can be used to derive soil temperatures from the air temperatures, but the effect of this on the simulation results needs to be investigated.
- Overall complexity of the model, particularly the biological aspect, makes modification and maintenance of the model cumbersome. Hence, while the developing simulation model provides a superb base for testing biological theories, it still fall short of being accurate enough to be used as a predictive

tool as part of a G.I.S. A primary objective of implementation using the G.I.S / UNIX platform was that it could interface directly with the climatic data assimilated by the ISCW. However, that data proved less useful that originally anticipated.

- The model ultimately needs to operate at more geographic locations simultaneously. In practice, the current model is equipped to do this. It is merely necessary to add a small amount of software code in order to allow weather data for extra sites to be loaded, and to accommodate the output matrices, as well as to copy and rename the existing lifecycle model code to represent the extra sites. Considerable economies can be realised if the same code is used, and the internal matrices named appropriately.
  Naturally, such multi-point simulations will run slower, but since the initiation of this project, significantly faster personal computers have become affordable.

- Wind data for testing the model has been obtained on diskette from the South African Weather Bureau. The fullness of these data are reflected in Figure 4.1. At the time of receiving these data (late 1994), it represented the most complete and up-to-date wind measurements that the SAWB could supply. Since that time, the SAWB has obtained the ETA model - developed by the American National Weather Service. This model came on-line in April of 1996, and has the potential to provide all the wind data required by the locust simulation model.

  Wind data obtainable from the ETA system covers an area from 6°s···54°s, 18°w···66°e, on a 49x85 grid. This is a substantially larger area of cover than is of interest to the locust modelling project, and spurious records can be ignored. The grid resolution is 1° north-south and east-west and the data represents *u* and *v* components 10 metres above the surface (sea or land). The predictive forecast is made every six hours, and corrected to match actual observations once every 24 hours.

  At the time of testing of the locust simulation model, no rainfall or air- or soil temperatures were available for any part of 1996. Thus the ETA model data was not useable. However, for future applications, it provides the most accurate, reliable and comprehensive wind measurements available.

- Accurate field data needs to be obtained, or made available in a suitable file format. The monthly cumulative totals available at present are not particularly helpful in evaluating or correcting timing errors in the simulated outbreaks. Attention should be given to correct timing of outbreaks before the question of magnitude accuracy is addressed. Admittedly measuring the magnitude of a real outbreak is considerably more difficult than merely noting the presence of an outbreak on a particular day - which is all that is required for accurate evaluation of simulation timing errors.

LIST OF REFERENCES

1. Allen, J. C. (1992) Nonlinear dynamics and chaos in insect population models. *Basics of insect modelling*. Ed. J. L. Goodenough & J. M. McKinion. American Society of Agricultural Engineers, St. Joseph, MI. pp. 23-34.

2. Antonov, A. G. & Kambulin, V. E. (1996) The season dynamics of the "Locusta migratoria migratoria phragmites australis" system computer forecasting. *Bulletin of the Kazakh Scientific Research Institute of Plant Protection*. Almaty, Rep of Kazakhstan. pp. 1-8.

3. Backlund, H. (1955) Red locusts and vegetation. *Oikos*. vol. 6. pp. 124-148.

4. Baker, P. S., Gewecke, M. & Cooter, R. J. (1984) Flight orientation of swarming *Locusta migratoria*. *Physiological entomology*. vol. 9. pp. 247-252.

5. Batschelet, E. (1981) *Circular statistics in biology*. Academic Press. London. pp. 279-282

6. Batten, A. (1967) Seasonal movements of swarms of *Locusta migratoria* in western Africa in 1928 to 1931. *Bulletin of entomological research*. 57 (iii). pp. 357-380.

7. Berryman, A. A. (1987) The theory and classification of outbreaks. *Insect outbreaks*. Ed. P. Barbosa & J. C. Schultz. Academic Press Inc. San Diego. pp. 3-26.

8. Botha, D. H. (1967) The viability of brown locust eggs [*Locustana pardalina* (Walker)]. *South African Journal of Agricultural Science*. vol. 10. pp 445-460.

9. Botha, D. H. & Jansen, A. (1969) Night flying by brown locusts, Locustana pardalina (Walker). *Phytophylactica*. vol. 1. pp. 79-92.

10. Bradley, J. V. (1968) *Distribution-free statistical tests*. Prentice-Hall. New Jersey. pp. 88-133, 284-302.

11. Brown, E. S. (1970) Nocturnal insect flight direction in relation to the wind. *Proceedings of the Royal Entomological Society*. London. vol. 45 (i-iii). pp. 39-43.

12. Brown, J. H. & Lomolino, M. V. (1998) Biogeography 2nd ed. Sinauer Associates, Sunderland MA. pp. 317-319.

13. Cheke, R. A. (1978) Theoretical rates of increase in gregarious and solitarious populations of the desert locust. *Oecologia*. vol. 35. pp. 161-171.

14. Clarke, D. P. (1968) Night flights of the Australian plague locust *Chortoicetes terminifera* Walk. in relation to storms. *Australian Journal of Zoology*. vol. 17. pp 329-352.

15. C[illegible]e, D. P. (1971) Flights after sunset by the Australian plague locust *Chortoicetes terminifera* (Walk.) and their significance in dispersal and migration. *Australian Journal of Zoology*. vol. 19. pp. 159-176.

16. Conover, W. J. (1971) *Practical nonparametric statistics*. Wiley & Sons. New York. pp. 309-316.

17. Davies, H. G. & Hicks, G. A. (1975) *Mathematics for scientific and technical students*. Longman. New York. pp. 312-327.

18. Day, R. K. & Knight, J. D. (1995) Operational aspects of forecasting migrant insect pests. In *Insect migration: tracking resources through space and time*. Ed. V. A. Drake & A. G. Gatehouse. Cambridge University Press. Cambridge. pp. 323-334.

19. Dennis, B., Kemp, W. P. & Beckwith, R. C. (1986) Stochastic model of insect phenology: estimation and testing. *Environmental Entomology*. vol. 15. pp. 540-544.

20. De Villiers, W. M. (1988) On the plague dynamics of the Brown Locust - *Locustana pardalina* (Walker). *Proceedings of the Locust Symposium*. Ed. B. McKenzie & M. Longridge. McGregor Museum. Kimberley. pp. 41-49.

21. Drake, V. A. & Farrow, R. A. (1983) The nocturnal migration of the Australian plague locust, *Chortoicetes terminifera* (Walker); quantitative radar observations of a series of northward flights. *Bulletin of entomological research*. vol. **173**. pp. 567-585.

22. Drake, V. A. & Farrow, R. A. (1988) The influence of atmospheric structure and motions on insect migration. *Annual review: Entomology*. vol. **33**. pp. 183-210.

23. Drake, V. A., Gatehouse, A. G. & Farrow, R. A. (1995) Insect migration: a holistic conceptual model. In *Insect migration: tracking resources through space and time*. Ed. V. A. Drake & G. A. Gatehouse. Cambridge University Press. Cambridge. pp. 427-457.

24. du Plessis, C. (1938) The influence of weather conditions on the incipient swarming of the brown locust *Locustana pardalina* (Walker). *Dept. of Agriculture and Forestry Science Bulletin* no. **186**. pp. 5-50.

25. Farrow, R. A. (1990) Flight and migration in Acridoids. In *Biology of grasshoppers*. Ed. F. Chapman & A. Joern. John Wiley. New York. pp. 227-313.

26. Farrow, R. A. & Longstaff, B. C. (1986) Comparison of the annual rates of increase of locusts in relation to the incidence of plagues. *Oikos*. Vol. **46**. pp. 207-222.

27. Faure, J. C. (1932) The phases of locusts in South Africa. *Bulletin of Entomological Research* Vol. **23**. pp. 293-405.

28. Fisher, N. I. (1993) *Statistical analysis of circular data*. Cambridge University Press. Cambridge. pp. 48-57.

29. Hanrahan, S. A. (1988) The locust problem and integrated pest management. *South African Journal of Science*. vol. **84**. pp. 5-8.

30. Hanrahan, P. (1994) Personal Communication. Department of Zoology, University of the Witwatersrand, Private Bag 3, WITS, 2050.

31. Hanrahan, P. & Hanrahan, S. A. (1994) *Developing an early warning system to predict Brown Locust outbreaks - report on phase 2 - computerisation of locust simulation model*. Unpublished.

32. Hanrahan, S. A. (1998) Personal Communication. Department of Zoology, University of the Witwatersrand, Private Bag 3, WITS, 2050.

33. Higgins, K., Hastings, A., Sarvela, J. N. & Botsford, L. W. (1997) Stochastic dynamics and deterministic skeletons: population behaviour of Dungeness crab. *Science*. vol. **276**. pp. 1431-1434.

34. Hill, D. S. (1983) *Agricultural insect pests of the tropics and their control*. 2nd ed. Cambridge University Press. Cambridge. pp. 119-162.

35. Hendrie, L. K., Irwin M. E., Liquido, N. J., Ruesin, W. G, Mueller, E. A., Voegtlin D. J., Achtmeier, G. L., Steiner, W. M. & Scott, R. W. (1985) Conceptual approach to modelling aphid migration, *The movement and dispersal of agriculturally important biotic agents*. Ed: D. R. Mackenzie, C. S. Barfield, G. G. Kennedy & R. D. Berger. Claitors, Baton Rouge. pp. 541-577.

36. Heyns, T. (1996) [illegible]nal Communication. Institute for Soil, Climate and Water. Private Bag X79, Pretoria.

37. Hollander, M. [illegible]e, D. A. (1973) *Non parametric statistical methods*. Wiley & Sons. New York. pp. 219-[illegible]

38. Hunter, D. M. (1981) Inhibition of daytime flight as a prerequisite for night migration in the Australian plague locust. *Research report (1981-82): Australian Plague Locust Commission*. Department of Primary Industry, Canberra. pp 17-20.

39. Kendall, M. G. (1962) *Rank correlation methods*. (3rd ed.) Griffin & Co. London. pp. 19-48.

40. Kisimoto, R. & Sogawa, K. (1995) Migr[illegible]n of the Brown Planthopper *Nilaparvata lugens* and the White-backed Planthopper *Sogatella furcifera* in East Asia: the rôle of weather and climate. In *Insect migration: tracking resources through space and time*. Ed. V. A. Drake & A. G. Gatehouse. Cambridge University Press. Cambridge. pp. 67-92.

41. Krebbs, C. J. (1978) A review of the Chitty Hypothesis of population regulation. *Journal Canadien de Zoologie*. vol. 56 (12). pp. 2464-2478.

42. Lambert, R. K. (1972) Some factors affecting flight in field populations of the Australian Plague Locust, *Chortoicetes terminifera*, in New South Wales. *Animal Behaviour*. vol. 20. pp. 205-217.

43. Lea, A. (1958) The continuing challenge of the brown locust numbers. *Journal of the Entomological Society of South Africa*. vol. 21. pp. 3-15.

44. Lea, A. (1968) Natural regulation and artificial control of brown locust numbers. *Journal of the Entomological Society of South Africa*. vol. 31. pp. 97-112.

45. Lecoq, M. (1995) Forecasting systems for migrant pests III - Locusts and grasshoppers in west Africa and Madagascar. *In Insect migration: tracking resources through space and time*. Ed. V.A. Drake & A. G. Gatehouse. Cambridge University Press. Cambridge. pp. 377-398.

46. Lehmann, E. L. & D'Abrera, H. J. M. (1975) *Nonparametrics: statistical methods based on ranks*. Holden-Day Inc. San Francisco. pp. 32-39, 164-174.

47. *Matlab reference guide* (1992) The Mathworks Inc. Natick, MA. pp. 27-548.

48. Matthèe, J. J. (1951) The structure and physiology of the egg of *Locustana pardalina* (Walk.). *Department of Agriculture Science Bulletin* vol. 316. Union of South Africa Government Printer. pp. 3-28.

49. Matthèe, J. J. (1961) Temperatuur as 'n mortaliteidsfaktor by die eier van die bruinsprinkaan, Locustana pardalina (Walk.). Source unknown, photocopy provided by Agromet/ISCW, Pretoria. pp. 629-640.

50. May, R. M. (1976) Models for single populations. *Theoretical Ecology: Principles and Applications*. Ed. R. M. May. Blackwell scientific. Oxford. pp. 5-27.

51. McCann, I. R., McFarland, M. J. & Witz, J. A. (1992) Near surface soil temperature model for biophysical development models. *Basics of insect modelling*. Ed. J. L. Goodenough & J. M. McKinion. American Society of Agricultural Engineers. pp. 94-104.

52. McKinion, J. M. (1992) Getting started: basics of modelling strategies. *Basics of insect modelling*. Ed. J. L. Goodenough & J. M. McKinion. American Society of Agricultural Engineers. St Joseph, Mi. pp. 1-6.

53. Moon, F. C. (1987) *Chaotic vibrations*. New York. Wiley & Sons. pp. 44-185.

54. Nailand P. & Hanrahan, S. A. (1993) Modelling brown locust, *Locustana pardalina* (Walker) outbreaks in the Karoo. *South African Journal of Science*. vol. 89. pp 420-424.

55. Napier-Backs, P. (1996) Personal Communication. Plant Protection Research Institute. Private Bag X134, Pretoria, 0001.

56. Neave, H. R. & Worthington, P. L. (1988) *Distribution-free tests*. Unwin Hyman. Boston. pp. 89-100, 69-75, 109-119, 149-184.

57. Pedgley, D. E. (1979) Weather during desert locust plague upsurges. *Phil. trans. of the Royal Society of London*. vol. 287. pp 387-391.

58. Pick, F. E. & Lea, A. (1970) Field observations on spontaneous movements of solitarious hoppers of the Brown Locust, *Locustana pardalina* (Walker), and behavioural differences between various colour forms. *Phytophylactica*. vol. 2. pp. 203-210.

59. Piper, S. (1998) Personal Communication. Department of Zoology. University of Natal.

60. Price, R. E. (1988) The lifecycle of the brown locust with reference to egg viability. In *Proceedings of the locust symposium*. McKenzie, B. & Longridge, M. McGregor museum. Kimberley. pp. 27-40

61. Price, R. E. (1996) Personal Communication. Plant Protection Research Institute. Private Bag X134, Pretoria, 0001.

62. Rainey, R. C. (1951) Weather and the movements of locust swarms: a new hypothesis. *Nature*. vol. 168. p1057.

63. Rainey, R. C. (1973) Airborne pests and the atmospheric environment. *Weather*. vol. 28 (6). pp. 224-239.

64. Rainey, R. C. (1989) *Migration and meteorology: flight behaviour and the atmospheric environment of locusts and other migrant pests.* Oxford University Press. Oxford. pp. 11-87, 282-287.

65. Riley, J. R. & Reynolds, D. R. (1986) Orientation at night by high-flying insects. *Insect flight: Dispersal and migration*. Ed. W. Danthanarayana. Springer-Verlag. Berlin. pp.71-87.

66. Risch, S. J. (1987) Agricultural ecology and insect outbreaks. In *Insect outbreaks*. Ed. P. Barbosa & J. C. Schultz. Academic Press Inc. San Diego. pp. 217-238

67. Robinson, T. P. (1995) Geographic information systems and remotely sensed data for determining seasonal distribution of habitats of [illegible] nt pests. *In Insect migration: tracking resources through space and time*. Ed. V. A. Drake & A. [illegible]. Gatehouse. Cambridge University Press. Cambridge. pp. 335-352.

68. Scott, R. W. & Achtemeier, G. L. (1987) Estimating pathways of migrating insects carried in atmospheric winds. *Environmental entomology*. vol. 16. no 6, pp. 1244-1254.

69. Smit, C. J. B. (1939) Field observations on the brown locust in an outbreak area. *Science Bulletin*, Department of Agriculture of the Union of South Africa. vol. 6. pp 1-48.

70. Smit, C. J. B. (1941) Forecasts of incipient outbreaks of the Brown Locust. *The Journal of the Entomological Society of South Africa*. vol. IV. pp 200-221.

71. Starfield, A. M., Smith, K. A. & Bleloch, A. M. (1994) *How to model it - problem solving for the computer age*. McGraw Hill. New York. pp 1-14.

72. Stremmler, F. G. (1982) *Introduction to communication systems*. 2nd ed. Addison-Wesley. Reading MA. pp. 431-490.

73. Taylor, R. A. J. (1979) A simulation model of locust migratory behaviour. *Journal of Animal Ecology*, vol. 48. pp 557-602.

74. Thunholm, B. (1990) A comparison of measured and simulated soil temperature using air temperature and soil surface energy balance as boundary conditions. *Agricultural and forest meteorology*. vol. 53. pp. 59-72.

75. Toms, R. B. (1988) *General biology of the Brown Locust*. Report on orthoptera research. Dept. of General Entomology, Transvaal Museum, Pretoria. pp. 12-25.

76. Uvarov, B. P. (1928) *Grasshoppers and locusts: a handbook for their study and control*. Imperial Bureau of Entomology. London. pp. 273-278.

77. Uvarov, B. P. (1966) *Grasshoppers and locust: a handbook of general Acridology*. Vol. 1. Cambridge University Press. Cambridge. pp. 489-454.

78. Uvarov, B. P. (1977) *Grasshoppers and locusts: a handbook of general Acridology*, Vol. 2. Centre for Overseas Pest Research. London. pp. 524-531.

79. Wagner, T. L. & Willers, J. L. (1992) The rôle and relationship of the database with insect population models. In *Basics of insect modelling*. Ed. J. L. Goodenough & J. M. McKinion. American Society of Agricultural Engineers. St Joseph, MI. pp. 9-18.

80. Wantanabe, T. (1995) Forecasting systems for migrant pests II - the rice planthoppers *Nilaparvata lugens* and *Sogatella furcifera* in Japan. In *Insect migration: tracking resources through space and time*. Ed. V. A Drake & A. G. Gatehouse. Cambridge University Press. Cambridge. pp. 365-376.

81. Wiese, I. H. & Bot, J. (1973) Insecticides: their hazards and control. *Entomological memoirs*. vol. **30**. pp. 1-13.

82. Wolda, H. & Dennis, B. (1993) Density dependence tests, are they? *Oecologia*. vol. **95**. pp. 581-591.

83. Worner, S. P. (1991) Use of models in applied entomology: the need for perspective. *Environmental Entomology*. vol. **20 (3)**. pp. 768-773.

84. Wright, D. E. (1981) The origin and development of plagues. *Research report (1981-82): Australian Plague Locust Commission*. Department of Primary Industry, Canberra. pp 31-36.

85. Wright, D. E. (1987) Analysis of the development of major plagues of the Australian Plague Locust, *Chortoicetes terminifera* (Walker) using a simulation model. *Australian Journal of Ecology*. vol. **12**. pp. 423-437.

86. Zalom, F., Goodell, P. B., Lloyd, T. W., Barnett, W. W. & Bentley, W. J. (1983) *Degree-days: The calculation and use of heat units in pest management*. University of California Co-operative extension, Division of Agriculture and Natural Resources, Berkeley. pp. 2-10.

Appendix A Simulation test results

The following twelve simulation result plots for Douglas, De Aar, Graaff-Reinet and Pofadder for the period March 1986 through December 1989 are included for reference purposes.

The results are shown in 12 groups of four graphs per page. The two columns represent the two stations at which the simulation was performed; the top row showing simulated results, the bottom row showing actual control data.

Simulated result for site A (log_{10} numbers of swarms & hopper bands)	Simulated result for site B (log_{10} numbers of swarms & hopper bands)
Actual control data for site A (hopper bands & swarms combined)	Actual control data for site B (hopper bands & swarms combined)

Each two-site simulation is shown both with migration disabled, and with migration enabled, on separate pages.

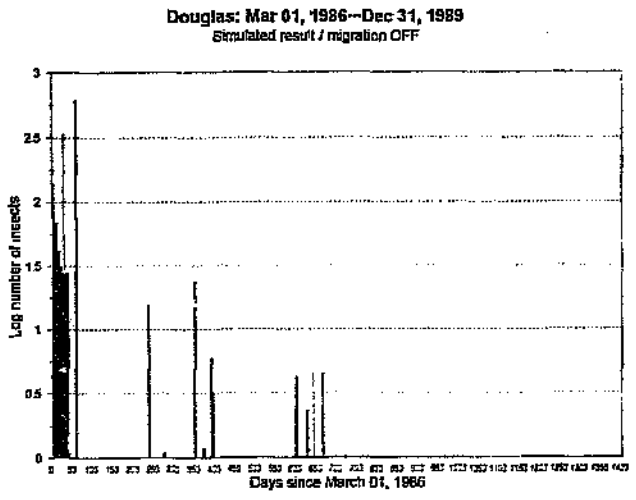
In each case the vertical axis represent total number of swarms and hopper bands combined (both simulated and field observations). The actual counts refers to the number of swarms or hopper bands controlled (poisoned).

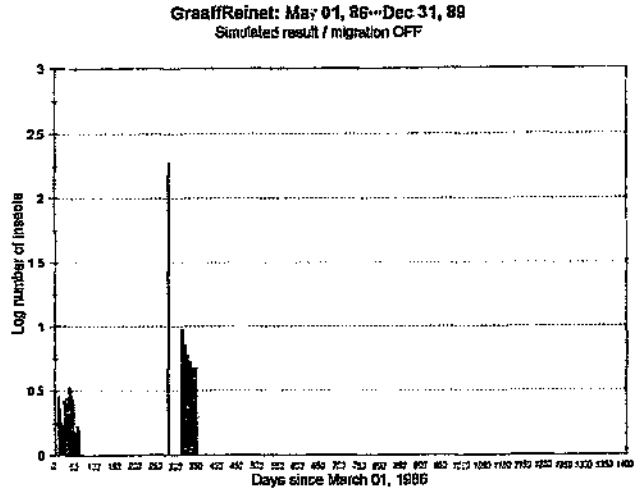
Control data represents monthly cumulative totals, whereas actual simulated results are shown for the day on which the model records such populatation fluctuations.

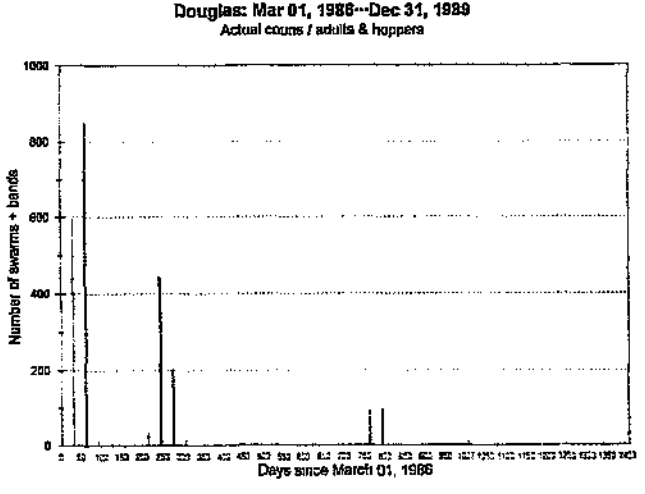
In each graph, the horizontal time axis represents days from the start of the simulation: March 1st, 1986.

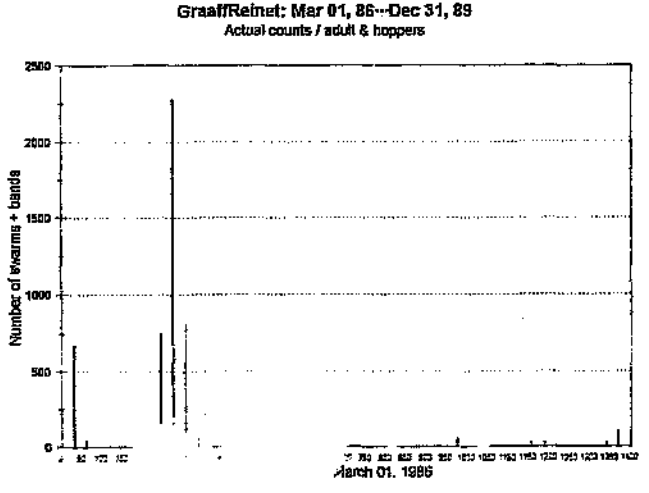
The sequence of simulations is as follows:

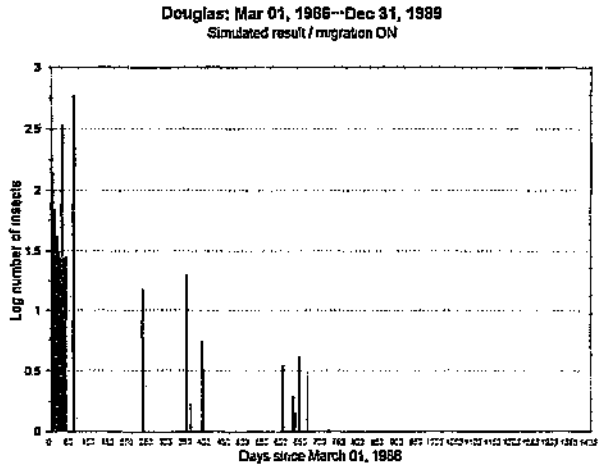
Douglas	⇔	Graaff-Reinet
Douglas	⇔	Pofadder
Douglas	⇔	De Aar
Graaff-Reinet	⇔	Pofadder
Graaff-Reinet	⇔	De Aar
Pofadder	⇔	De Aar

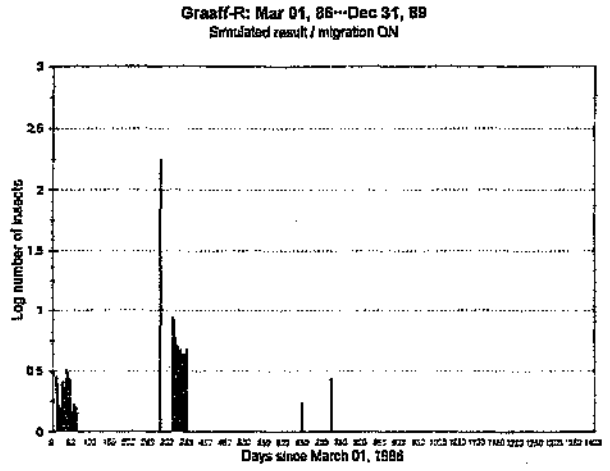

Douglas: Mar 01, 1986--Dec 31, 1989
Simulated result / migration OFF
Log number of insects
Days since March 01, 1986

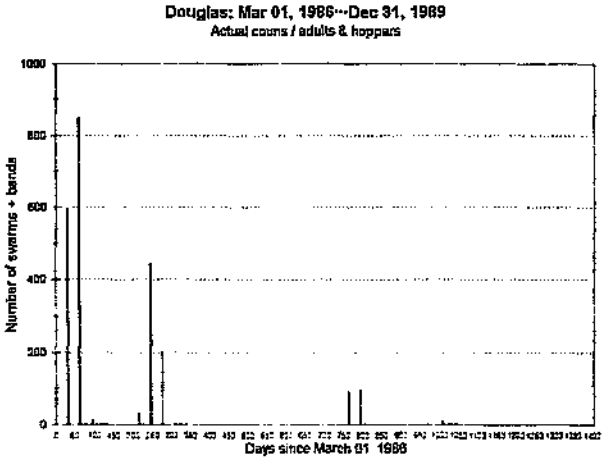

GraaffReinet: Mar 01, 86--Dec 31, 89
Simulated result / migration OFF
Log number of insects
Days since March 01, 1986

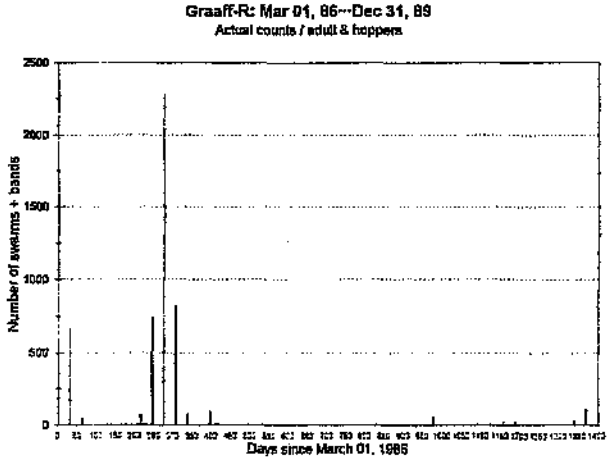

Douglas: Mar 01, 1986--Dec 31, 1989
Actual couns / adults & hoppers
Number of swarms + bands
Days since March 01, 1986

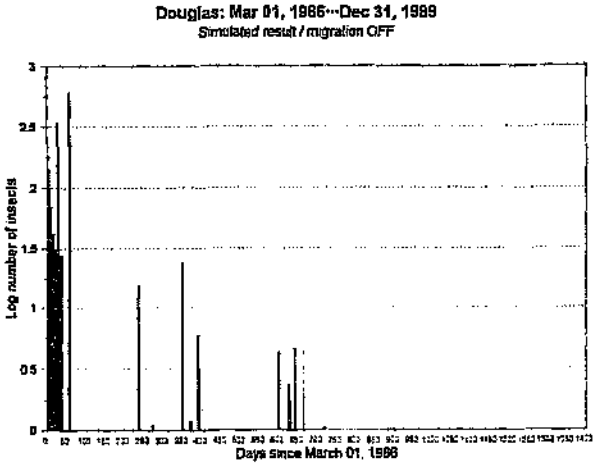

GraaffReinet: Mar 01, 86--Dec 31, 89
Actual counts / adult & hoppers
Number of swarms + bands
March 01, 1986

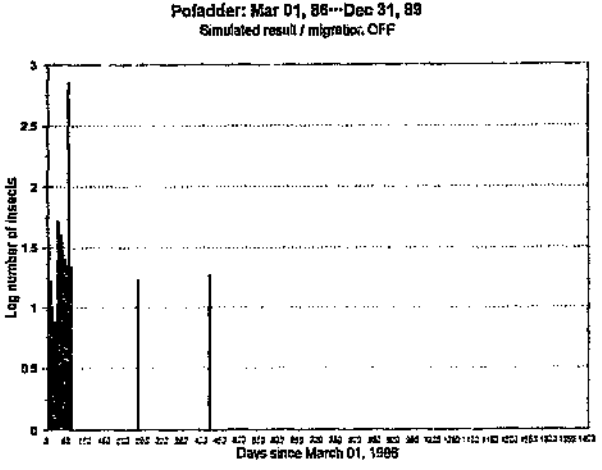

Douglas: Mar 01, 1986···Dec 31, 1989
Simulated result / migration ON
Log number of insects
Days since March 01, 1986

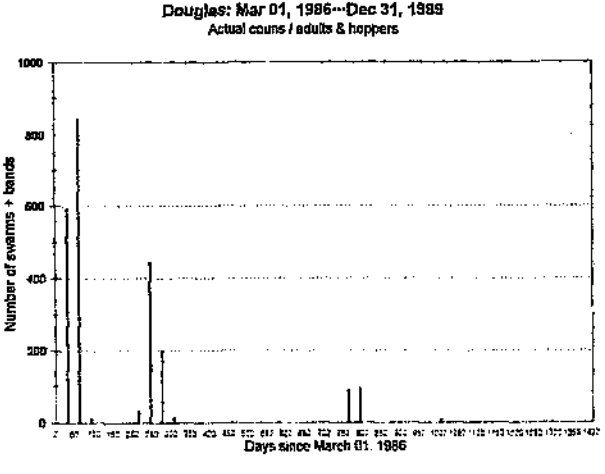

Graaff-R: Mar 01, 86···Dec 31, 89
Simulated result / migration ON
Log number of insects
Days since March 01, 1986

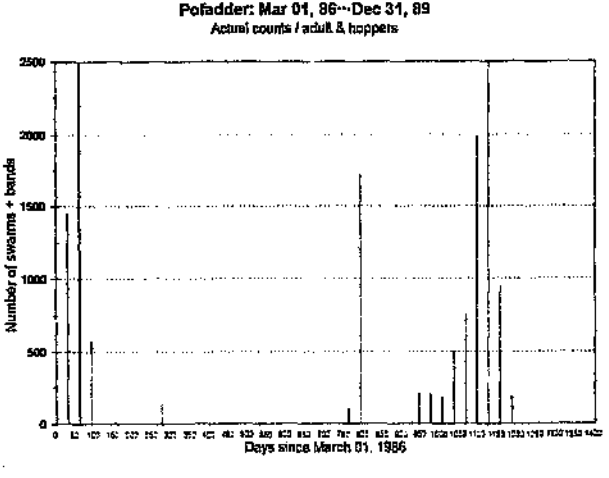

Douglas: Mar 01, 1986···Dec 31, 1989
Actual couns / adults & hoppers
Number of swarms + bands
Days since March 01 1986

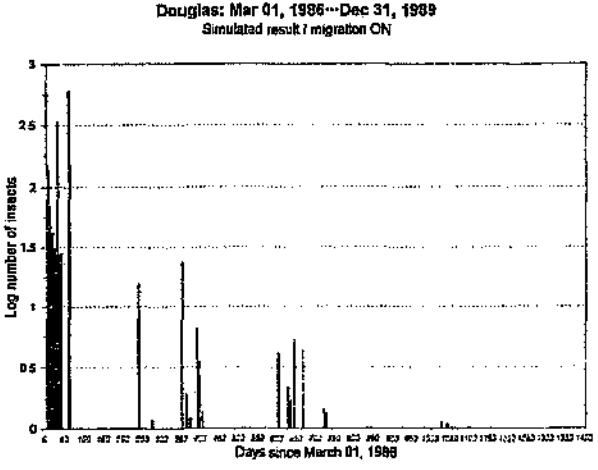

Graaff-R: Mar 01, 86···Dec 31, 89
Actual counts / adult & hoppers
Number of swarms + bands
Days since March 01, 1986

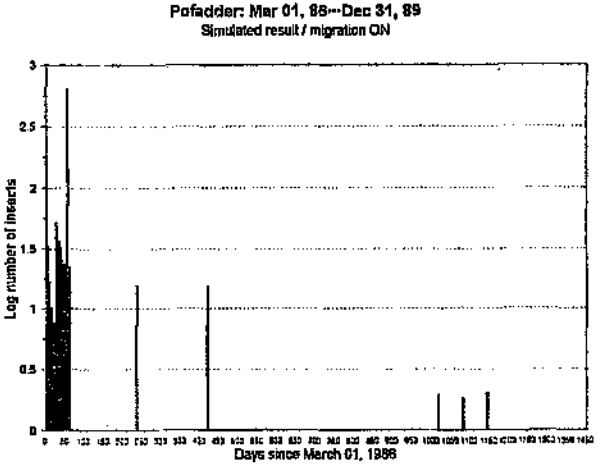

Douglas: Mar 01, 1986···Dec 31, 1989
Simulated result / migration OFF
Log number of insects
Days since March 01, 1986

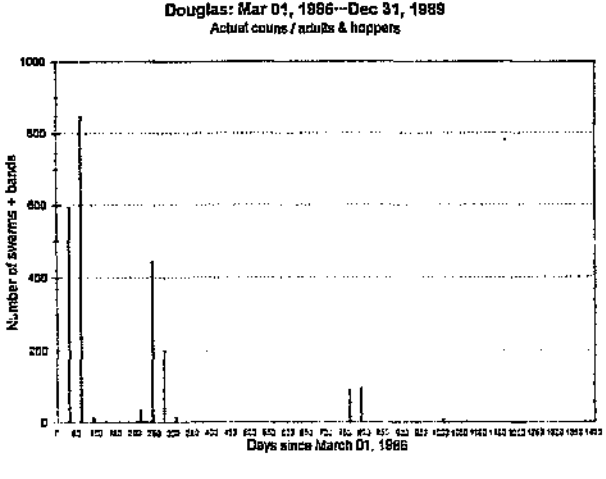

Pofadder: Mar 01, 86···Dec 31, 89
Simulated result / migration OFF
Log number of insects
Days since March 01, 1986

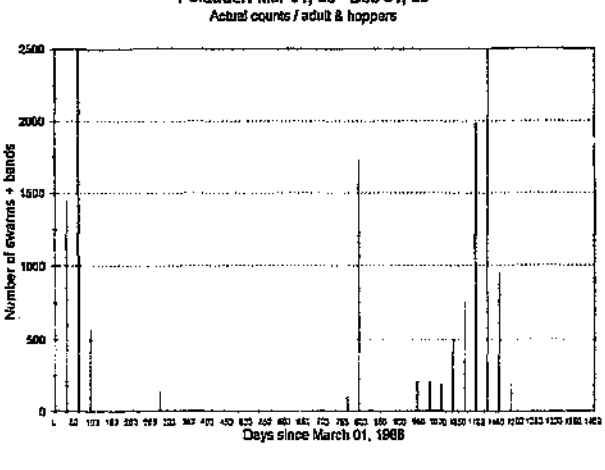

Douglas: Mar 01, 1986···Dec 31, 1989
Actual couns / adults & hoppers
Number of swarms + bands
Days since March 01, 1986

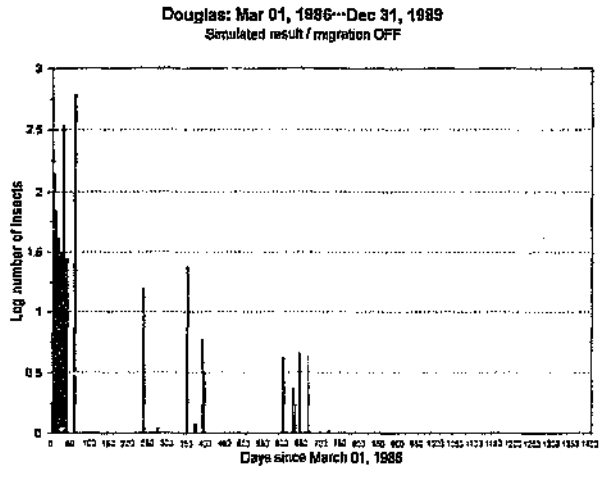

Pofadder: Mar 01, 86···Dec 31, 89
Actual counts / adult & hoppers
Number of swarms + bands
Days since March 01, 1986

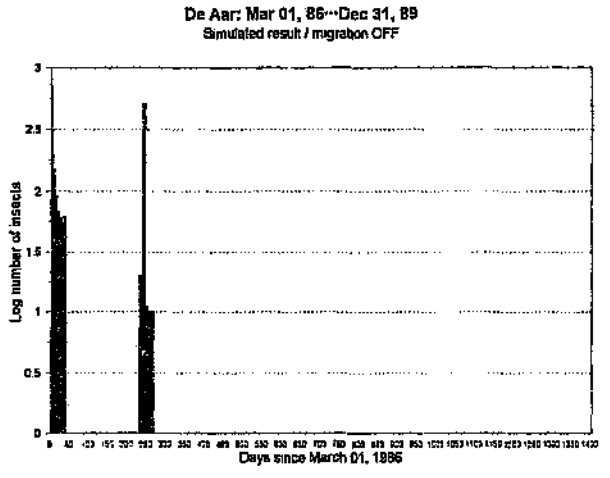

Douglas: Mar 01, 1986···Dec 31, 1989
Simulated result / migration ON
Log number of insects
Days since March 01, 1986

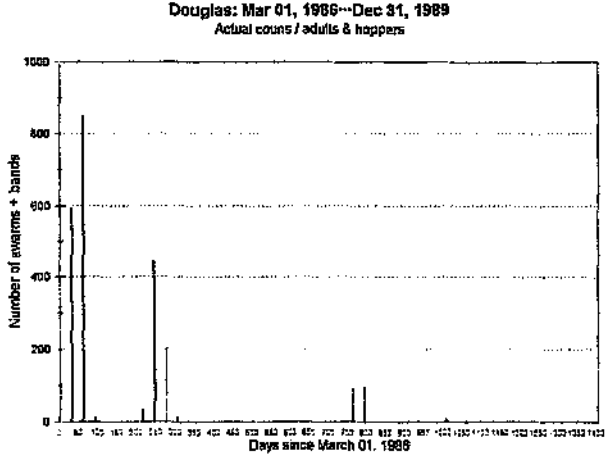

Pofadder: Mar 01, 86···Dec 31, 89
Simulated result / migration ON
Log number of insects
Days since March 01, 1986

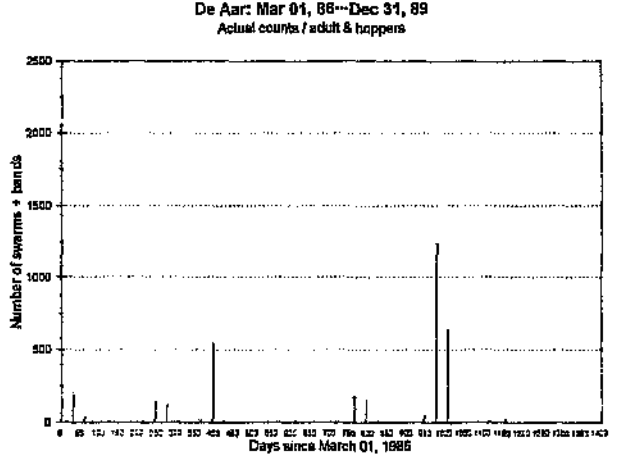

Douglas: Mar 01, 1986···Dec 31, 1989
Actual couns / adults & hoppers
Number of swarms + bands
Days since March 01, 1986

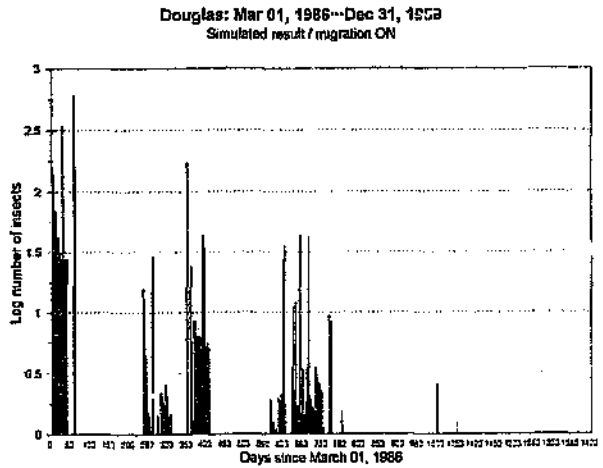

Pofadder: Mar 01, 86···Dec 31, 89
Actual counts / adult & hoppers
Number of swarms + bands
Days since March 01, 1986

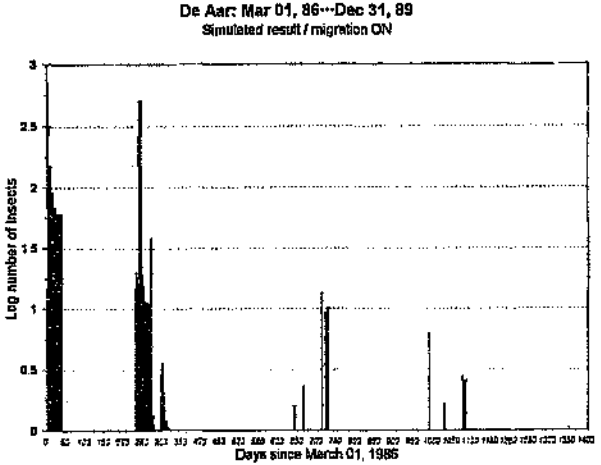

Douglas: Mar 01, 1986···Dec 31, 1989
Simulated result / migration OFF
Log number of insects
Days since March 01, 1986

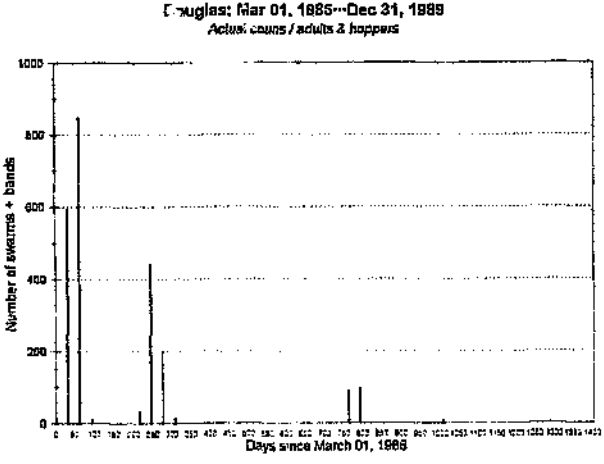

De Aar: Mar 01, 86···Dec 31, 89
Simulated result / migration OFF
Log number of insects
Days since March 01, 1986

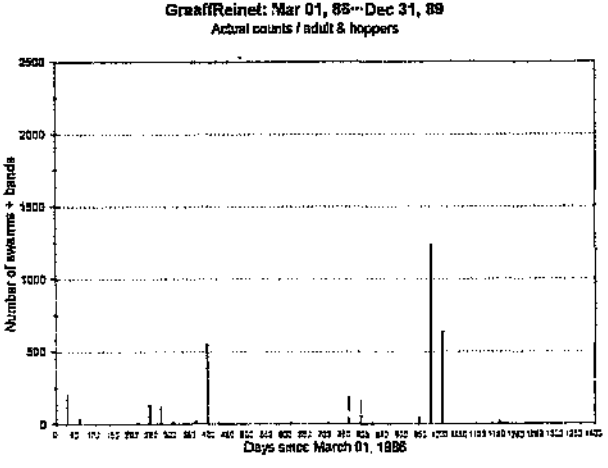

Douglas: Mar 01, 1986···Dec 31, 1989
Actual couns / adults & hoppers
Number of swarms + bands
Days since March 01, 1986

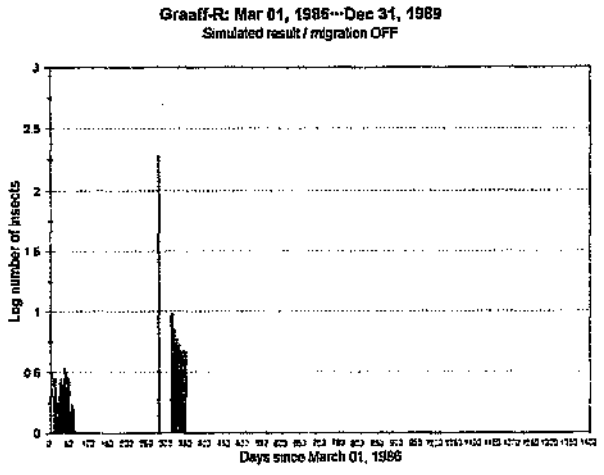

De Aar: Mar 01, 86···Dec 31, 89
Actual counts / adult & hoppers
Number of swarms + bands
Days since March 01, 1986

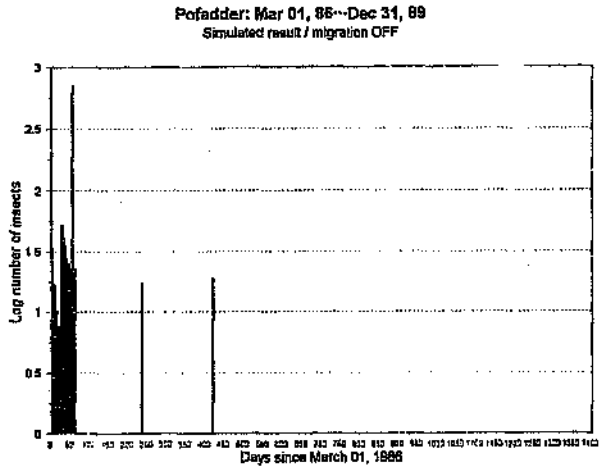

Douglas: Mar 01, 1986···Dec 31, 1989
Simulated result / migration ON
Log number of insects
Days since March 01, 1986

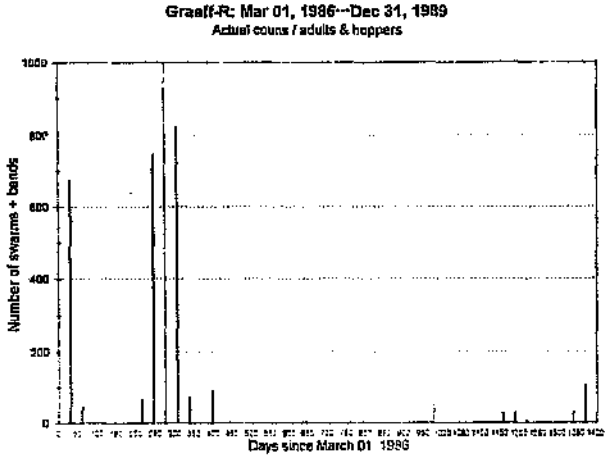

De Aar: Mar 01, 86···Dec 31, 89
Simulated result / migration ON
Log number of insects
Days since March 01, 1986

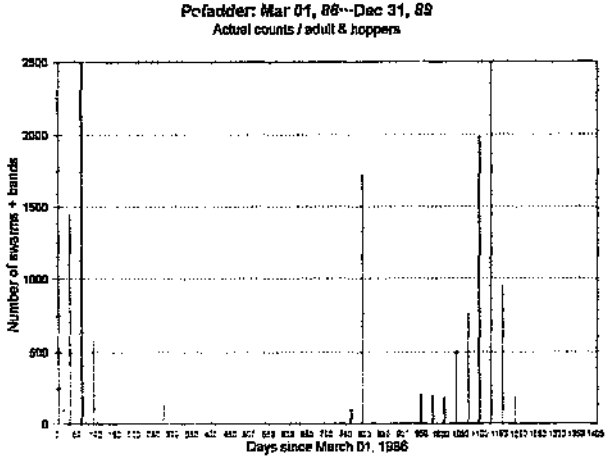

Douglas: Mar 01, 1986···Dec 31, 1989
Actual counts / adults & hoppers
Number of swarms + bands
Days since March 01, 1986

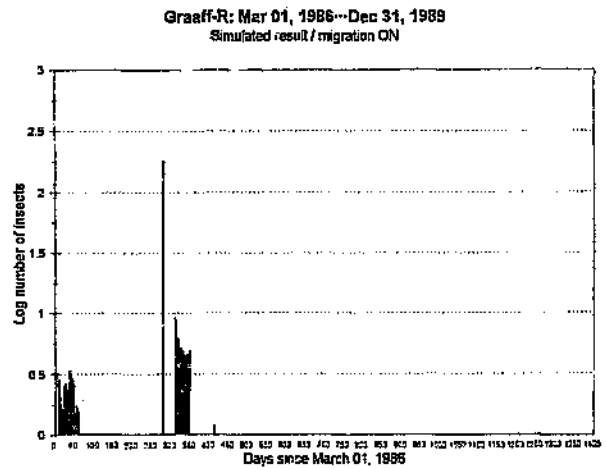

GraaffReinet: Mar 01, 86···Dec 31, 89
Actual counts / adult & hoppers
Number of swarms + bands
Days since March 01, 1986

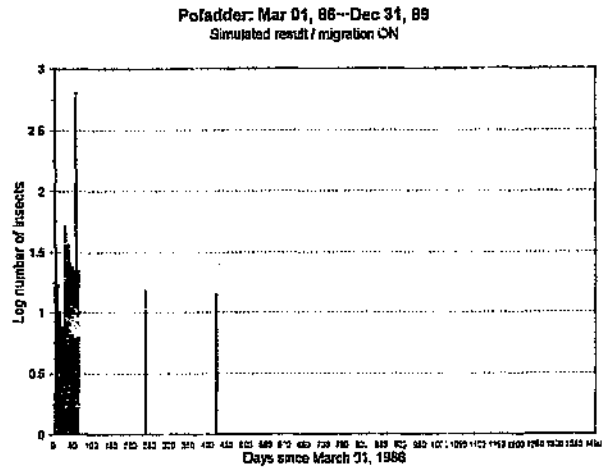

Graaff-R: Mar 01, 1986···Dec 31, 1989
Simulated result / migration OFF
Log number of insects
Days since March 01, 1986

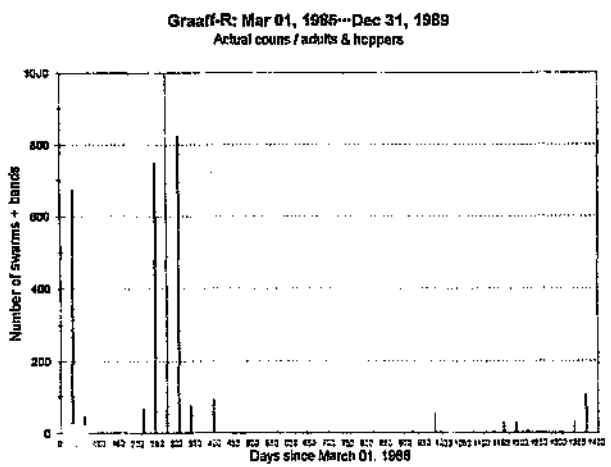

Pofadder: Mar 01, 86···Dec 31, 89
Simulated result / migration OFF
Log number of insects
Days since March 01, 1986

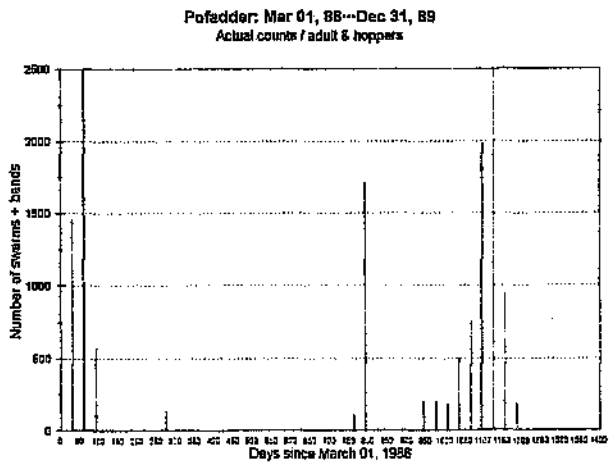

Graaff-R: Mar 01, 1986···Dec 31, 1989
Actual couns / adults & hoppers
Number of swarms + bands
Days since March 01 1986

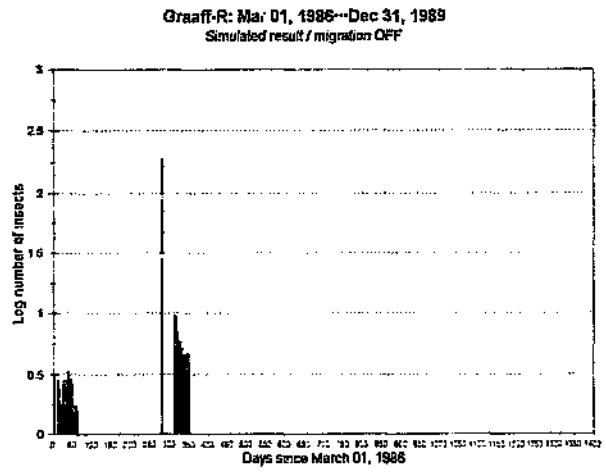

Pofadder: Mar 01, 86···Dec 31, 89
Actual counts / adult & hoppers
Number of swarms + bands
Days since March 01, 1986

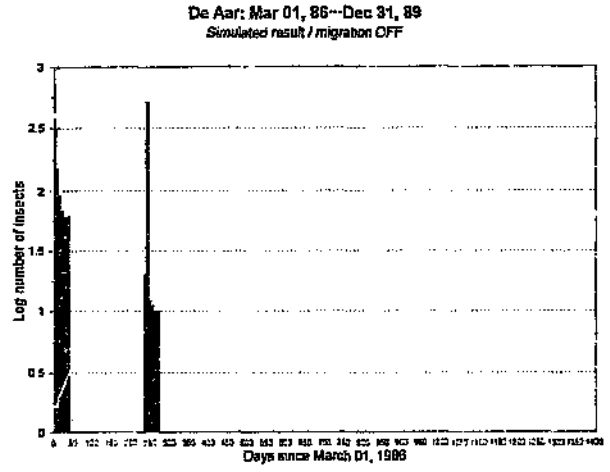

Graaff-R: Mar 01, 1986···Dec 31, 1989
Simulated result / migration ON
Log number of insects
Days since March 01, 1986

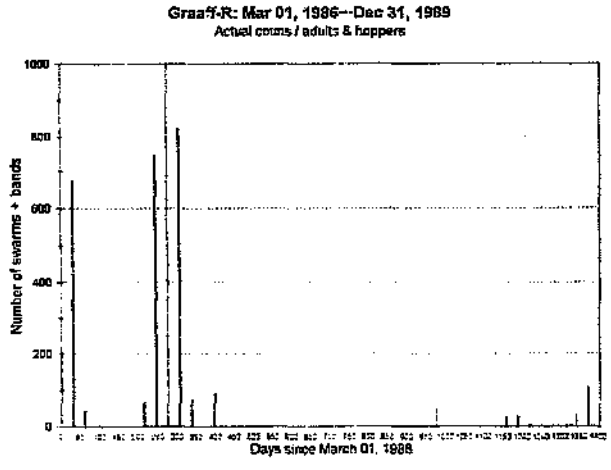

Pofadder: Mar 01, 86···Dec 31, 89
Simulated result / migration ON
Log number of insects
Days since March 01, 1986

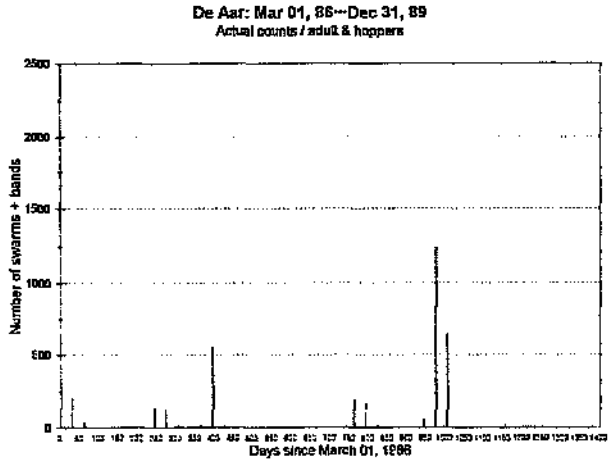

Graaff-R: Mar 01, 1986···Dec 31, 1989
Actual couns / adults & hoppers
Number of swarms + bands
Days since March 01, 1986

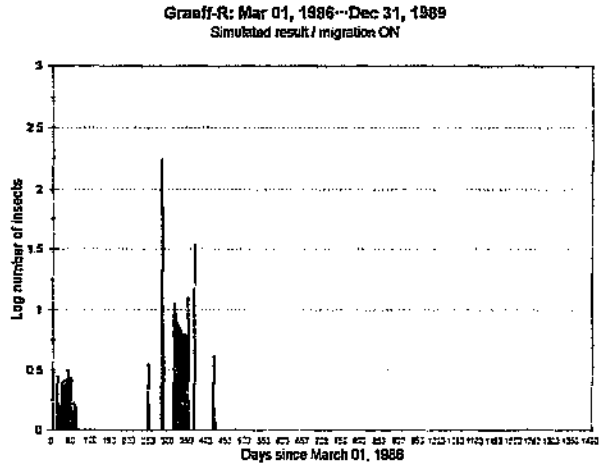

Pofadder: Mar 01, 86···Dec 31, 89
Actual counts / adult & hoppers
Number of swarms + bands
Days since March 01, 1986

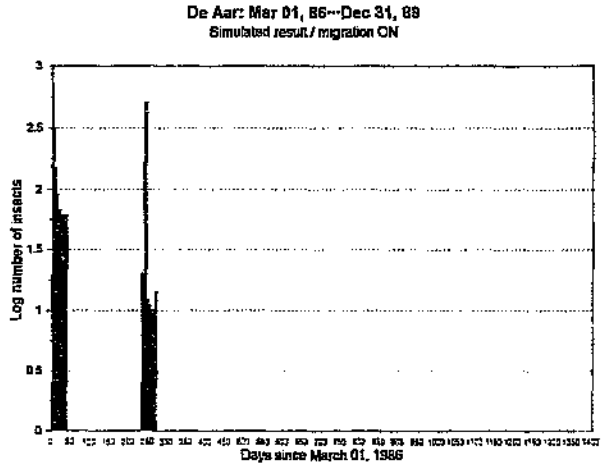

Graaff-R: Mar 01, 1986--Dec 31, 1989
Simulated result / migration OFF
Log number of insects
Days since March 01, 1986

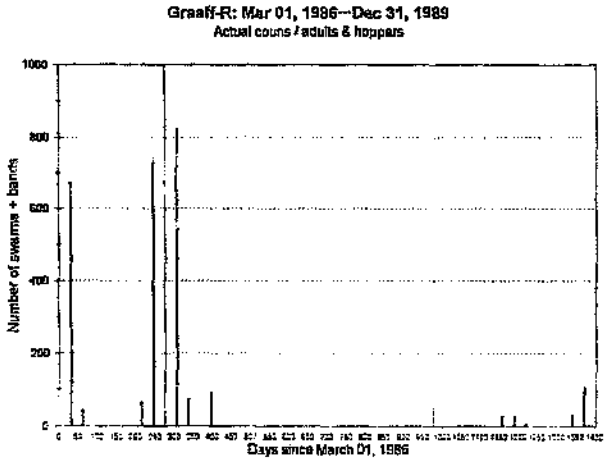

De Aar: Mar 01, 86--Dec 31, 89
Simulated result / migration OFF
Log number of insects
Days since March 01, 1986

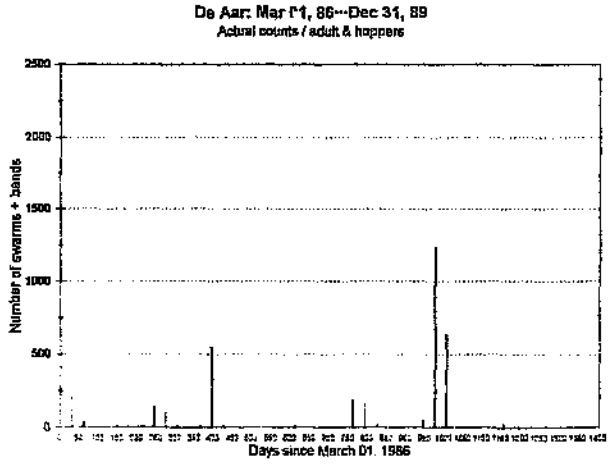

Graaff-R: Mar 01, 1986--Dec 31, 1989
Actual counts / adults & hoppers
Number of swarms + bands
Days since March 01, 1986

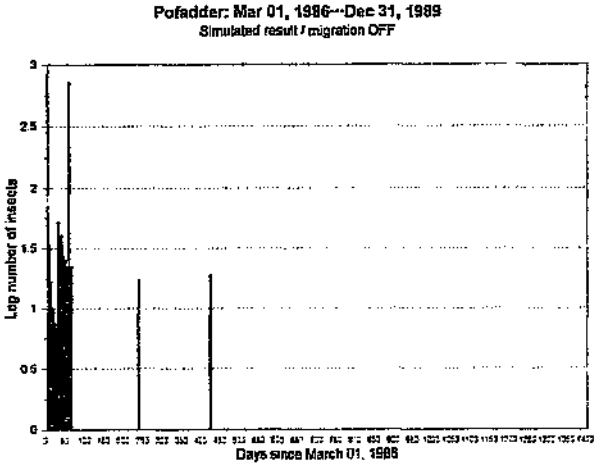

De Aar: Mar 01, 86--Dec 31, 89
Actual counts / adult & hoppers
Number of swarms + bands
Days since March 01, 1986

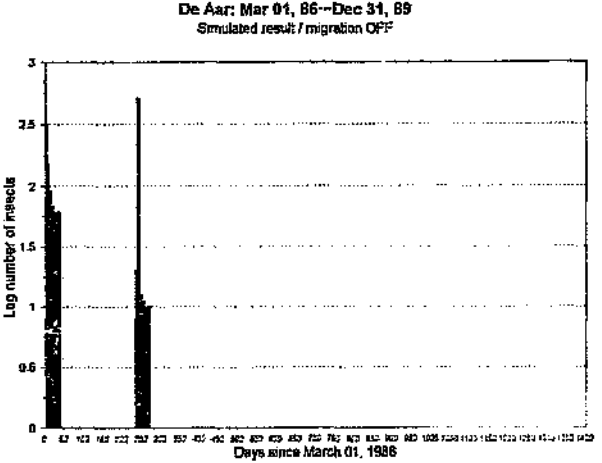

Graaff-R: Mar 01, 1986···Dec 31, 1989
Simulated result / migration ON
Log number of insects
Days since March 01, 1986

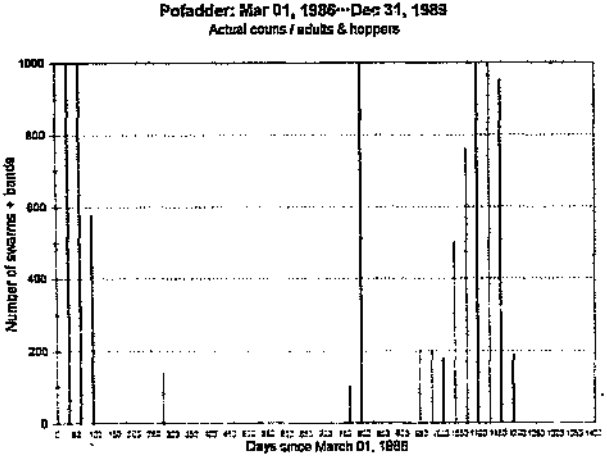

De Aar: Mar 01, 86···Dec 31, 89
Simulated result / migration ON
Log number of insects
Days since March 01, 1986

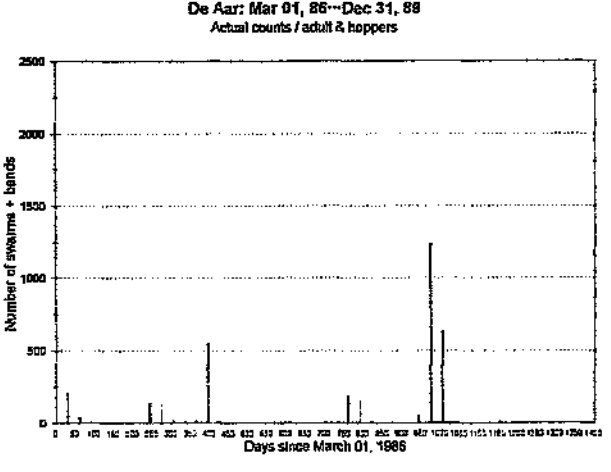

Graaff-R: Mar 01, 1986···Dec 31, 1989
Actual couns / adults & hoppers
Number of swarms + bands
Days since March 01, 1986

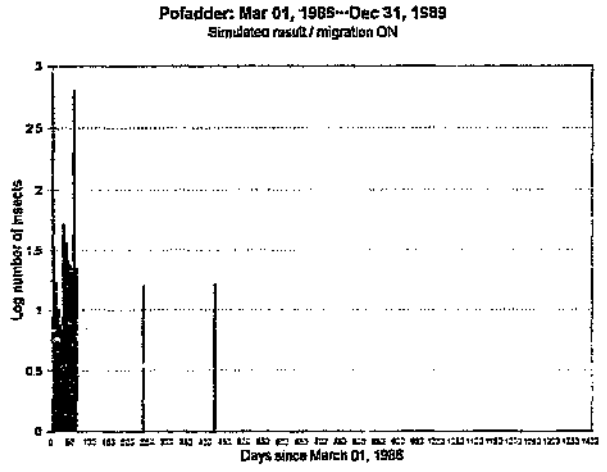

De Aar: Mar 01, 86···Dec 31, 89
Actual counts / adult & hoppers
Number of swarms + bands
Days since March 01, 1986

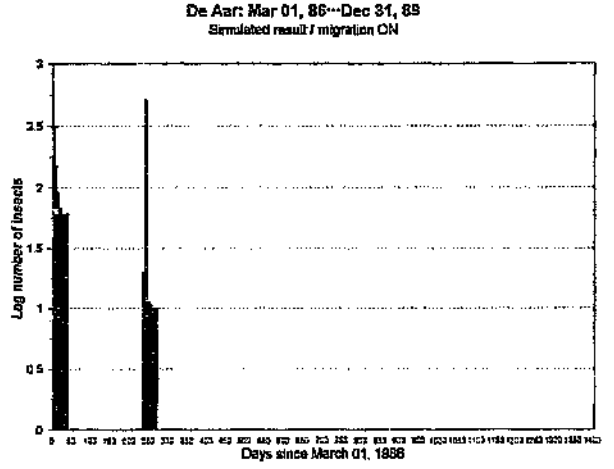

Pofadder: Mar 01, 1986···Dec 31, 1989
Simulated result / migration OFF
Log number of insects
Days since March 01, 1986

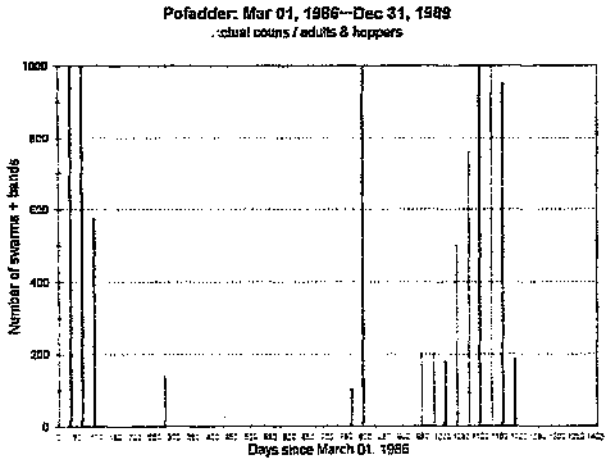

De Aar: Mar 01, 86···Dec 31, 89
Simulated result / migration OFF
Log number of insects
Days since March 01, 1986

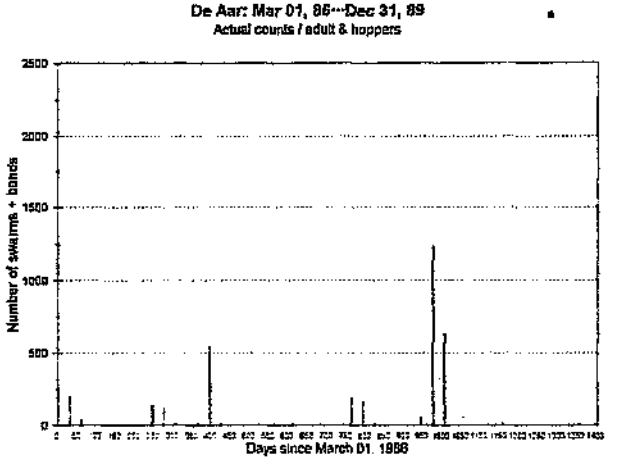

Pofadder: Mar 01, 1986···Dec 31, 1989
Actual couns / adults & hoppers
Number of swarms + bands
Days since March 01, 1986


De Aar: Mar 01, 86···Dec 31, 89
Actual counts / adult & hoppers
Number of swarms + bands
Days since March 01, 1986


Pofadder: Mar 01, 1986···Dec 31, 1989
Simulated result / migration ON
Log number of insects
Days since March 01, 1986


De Aar: Mar 01, 86···Dec 31, 89
Simulated result / migration ON
Log number of insects
Days since March 01, 1986

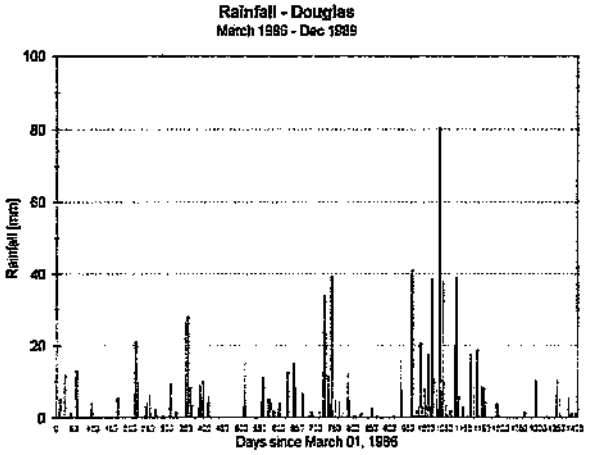

Pofadder: Mar 01, 1986···Dec 31, 1989
Actual counts / adults & hoppers
Number of swarms + bands
Days since March 01, 1986

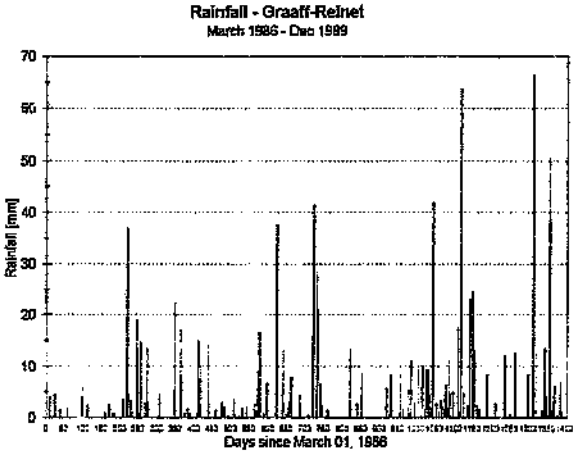

De Aar: Mar 01, 86···Dec 31, 89
Actual counts / adult & hoppers
Number of swarms + bands
Days since March 01, 1986

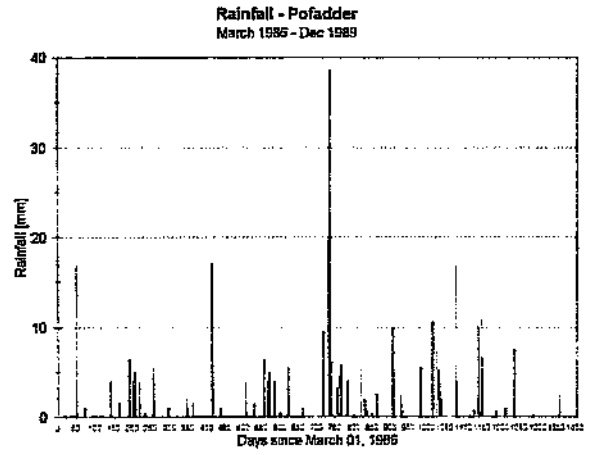
Appendix B Rainfall graphs for simulation period

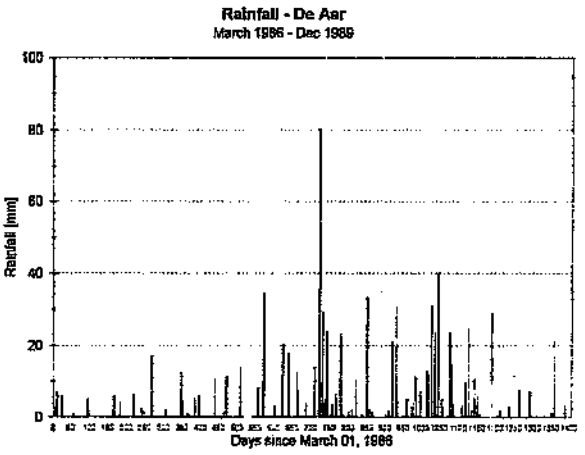
The following rainfall graph plots for Douglas, De Aar, Graaff-Reinet and Pofadder for the period March 1986 through December 1989 are included for reference purposes.

Note that the time axis represents days since March 1st, 1986, so that rainfall events maybe compared directly to those events on the simulation plots.


Rainfall - Douglas
March 1986 - Dec 1989
Rainfall [mm]
Days since March 01, 1986


Rainfall - Graaff-Reinet
March 1986 - Dec 1989
Rainfall [mm]
Days since March 01, 1986


Rainfall - Pofadder
March 1986 - Dec 1989
Rainfall [mm]
Days since March 01, 1986


Rainfall - De Aar
March 1986 - Dec 1989
Rainfall [mm]
Days since March 01, 1986

Appendix C Statistical correlation tests applied to locust populations, simulated and controlled (killed).

Population = combined hopper bands and adult swarms

Quantiles for the α=5% level of significance are shown for diferent sample sizes.

D=Kolmogorov-Smirnov	$W_p = 0{,}0513\,|_{n=1400}$	$W_p = 0{,}0466\,|_{n=1700}$
ρ=Pearson product moment	W_p = unknown, dependant on joint-PDF.	
R_S=Spearman rank	$W_p = 0{,}0440\,|_{n=1400}$	$W_p = 0{,}0399\,|_{n=1700}$
τ=normalised Kendall rank	$W_p = 0{,}0239\,|_{n=1400}$	$W_p = 0{,}0266\,|_{n=1700}$

NORMAL text denotes station A , *ITALIC* text denotes station B. Improvement or deterioration may be seen by comparing left column to right column directly.Station A is given first - the top-most group of figures.

WITHOUT MIGRATION	WITH MIGRATION
Test on simulation 1 Douglas ⇔ Graaff-Reinet	Test on simulation 1 (with migration enabled) Douglas ⇔ Graaff-Reinet
$D = 0.1598$ $\rho = 0.3457$ $R_s = 0.3276$ $\tau = 0.3697$	$D = 0.5877$ $\rho = 0.3457$ $R_s = 0.4919$ $\tau = 0.3958$
$D = 0.0271$ $\rho = 0.0456$ $R_s = 0.8286$ $\tau = 0.1726$	$D = 0.6434$ $\rho = 0.0456$ $R_s = 0.3462$ $\tau = 0.3613$
Test on simulation 2 Douglas ⇔ Pofadder	Test on simulation 2 (with migration enabled) Douglas ⇔ Pofadder
$D = 0.1598$ $\rho = 0.3457$ $R_s = 0.3276$ $\tau = 0.3697$	$D = 0.5093$ $\rho = 0.3457$ $R_s = 0.3579$ $\tau = 0.3129$
$D = 0.0257$ $\rho = -0.0040$ $R_s = 0.6904$ $\tau = 0.1391$	$D = 0.5214$ $\rho = -0.0042$ $R_s = 0.3968$ $\tau = 0.0610$
Test on simulation 3 Douglas ⇔ De Aar	Test on simulation 3 (with migration enabled) Douglas ⇔ De Aar
$D = 0.1392$ $\rho = 0.3458$ $R_s = 0.3405$ $\tau = 0.2799$	$D = 0.3945$ $\rho = 0.3107$ $R_s = 0.4230$ $\tau = 0.4092$
$D = 0.0057$ $\rho = 0.0046$ $R_s = 0.7910$ $\tau = 0.2595$	$D = 0.3401$ $\rho = 0.0073$ $R_s = 0.4688$ $\tau = 0.3041$

Test on simulation 4 Graaff-Reinet ⇔ Pofadder	Test on simulation 4 (with migration enabled) Graaff-Reinet ⇔ Pofadder
D = 0.0278	D = 0.8138
ρ = 0.0453	ρ = 0.0452
R_s = 0.8277	R_s = 0.4881
τ = 0.1738	τ = 0.3622
D = 0.0257	*D = 0.7753*
ρ = -0.0040	*ρ = -0.0042*
R_s = 0.6904	*R_s = 0.4772*
τ = 0.1391	*τ = 0.3574*

Test on simulation 5 Graaff-Reinet ⇔ De aar	Test on simulation 5 (with migration enabled) Graaff-Reinet ⇔ De aar
D = 0.0278	D = 0.4579
ρ = 0.0453	ρ = 0.0443
R_s = 0.8277	R_s = 0.3254
τ = 0.1738	τ = 0.4196
D = 0.0078	*D = 0.3652*
ρ = -8.2088e-004	*ρ = -9.1729e-004*
R_s = 0.7431	*R_s = 0.3723*
τ = 0.3182	*τ = 0.4836*

Test on simulation 6 Pofadder ⇔ De aar	Test on simulation 6 (with migration enabled) Pofadder ⇔ De aar
D = 0.0257	D = 0.4066
ρ = -0.0043	ρ = -0.0044
R_s = 0.6898	R_s = 0.3135
τ = 0.1402	τ = 0.3311
D = 0.0078	*D = 0.3609*
ρ = -8.2088e-004	*ρ = -8.9606e-004*
R_s = 0.7431	*R_s = 0.3761*
τ = 0.3182	*τ = 0.4039*

Appendix D Summary of weather data missing from data set used to perform test simulations.

The following list outlines those sites and dates within the weather data set used to test the model, that exhibited large absences of data values. In each case, the method used to interpolate the data is given.

SITE	INTERVAL	MISSING DATA	REPLACEMENT DATA
Douglas:	April 88	air °C, soil °C	April 86 records.
	April 88	rainfall	Zero, except where sudden drops in °C of air and soil. Used typical seasonal rainfall values.
Pofadder	Jan 85..Aug 85	air °C	Based on soil °C, using 1986 seasonal mean differences
	Jan 85..Aug 85	rainfall	Zero, except where sudden drops in °C of soil. Typical seasonal rainfall values used.
	1988	soil $°C_{max}$ (sparse)	Based on soil $°C_{min}$ using 1987 mean seasonal difference.
	Dec 88	soil °C	Based on air °C, using Dec '87 mean seasonal difference.
	1989	soil °C (sparse)	Based on air °C, using 1987 mean seasonal difference.
Graaff-Reinet	April 88	°C soil, °C air	Substituted April 1987 °C.
	Jul 88..Aug 88	soil $°C_{min}$	Based on soil $°C_{max}$ using 1987 mean seasonal difference.
	Dec 88	soil °C	Based on air °C using 1987 mean seasonal difference.
	Aug 88..Dec 89	soil °C	Based on air °C using 1987 mean seasonal difference.
De Aar	1985..1990	°C $soil_{min}$	Based on °C $soil_{max}$ using Graaff-Reinet mean seasonal difference for corresponding seasons.
	Apr 88	°C $soil_{max}$	Based on °C air_{max} using 1987 mean seasonal difference
	Oct 88..Dec 88	°C $soil_{max}$	Based on °C air_{max} using 1987 mean seasonal difference.

Other smaller data "holes" can be found by means of the colour coded fields in the weather data file used as input to the test simulation runs.

Appendix E Section 1 Simulation Software Technical Reference

Getting started

NOTE: The following documentation uses illustrations representing displays under Windows™ 3.1, using a 600x800x256c SVGA display. Windows '95 may appear slightly differently on screen. Your display may also differ slightly depending on your VGA card, monitor and your Windows set-up.

TYPOGRAPHICAL CONVENTIONS

bold	Anything that you must type exactly as it appears. For example, if you are asked to type dir \windows, you would type the bold characters exactly as printed.
Italic	Place-holders for information you must provide. For example, if you are asked to type copy *filename*, you would type the actual file's name instead of the term in italics. Italic type also introduces a new term, or keyword.
UPPERCASE	Directory names, filenames, and acronyms.
↵	This symbol represents the carriage return (CR) or "Enter" key
^	This symbol represents a Control character. The CTRL key must be held while the character is typed. For example ^C means that you should hold down the CTRL key and while doing so, touch the "c" key.
ALT	ALT refers to the Alternate function key. This is similar to the Control key. The ALT key must be held while the character is typed. For example ALT F means that you should hold down the ALT key and while doing so, touch the "f" key.
	This symbol denotes additional information for the advanced user. It is not essential to basic operation.
	Helpful hints or tips that apply to all Windows applications, not only the Simulation Model.

SYSTEM REQUIREMENTS

The simulation model runs exclusively under Mathworks™ Matlab version 4.2 or higher, and requires Simulink™ version 1.3 or higher. Both Matlab and Simulink require Microsoft Windows™ 3.1 or higher.

IBM OS/2™ offers full support of Windows applications without the need for actually running MS-Windows.

"Enhanced Mode" operation of MS-Windows requires an 80386 CPU, or an 80486 CPU or higher (DX-2, DX-4, Pentium™, P24). 4MB (minimum) of RAM is required (16MB recommended). Simulations incorporating large numbers of nodes (more

than ≈4) or that run for long periods (more than ≈5 years) will require 8MB, 16MB or 32MB of RAM for efficient execution. If you are using a network installation, a permanent local Windows swapfile will improve performance.

If you are using an 80386 based machine, an 80387 floating point processor is required. Matlab cannot run on an 80486/Pentium based machine unless an 80487 floating point processor has been installed.

It is assumed that the reader is familiar with basic DOS and Windows procedures, such as file copying, editing of ASCII text files, text-highlighting, drag-'n'-drop mouse operation. Please refer to your Windows and DOS manuals if you need information on these topics.

NOTE: Windows NOTEPAD will frequently be inadequate for editing the text files generated by the simulation. It is recommended that an editor such as the Norton Utilities Editor (NE.EXE), or the Norton Desktop DeskEdit, or DOS's EDIT be used instead.

THE MS-WINDOWS™ OPERATING SYSTEM SHELL

Matlab needs to have been installed on your machine and needs to be aware of the directory structure of your model files. See the Matlab Users' Guide if you are unsure about how to set up the correct operation environment for Matlab.

All the model and simulation files are written in Matlab's own interpretive language, and have the DOS file extension .M
These files should all reside under the same directory path, for example:
C:\WINDOWS\MATLAB\MODEL*filenames*.M
Windows' Program Manager should be made aware of this path by setting the file properties associated with the Locust Simulation icon to point to the correct start-up directory.

Windows™ 3.1 / 3.11 will run faster and more efficiently if you **do not** use a third party shell, such as Norton Desktop™. To remove Norton Desktop from your installation, edit the file SYSTEM.INI in your Windows directory (use Windows Notepad). Under the **[boot]** section of SYSTEM.INI, replace the instruction:
shell=ndw.exe with the instruction:
shell=progman.exe
Remember to save the file!

Once correctly installed, the simulation can be accessed by double-clicking the Locust Simulation program icon (a WITS University crest) displayed in the Matlab program group, as shown in Figure E1.1.

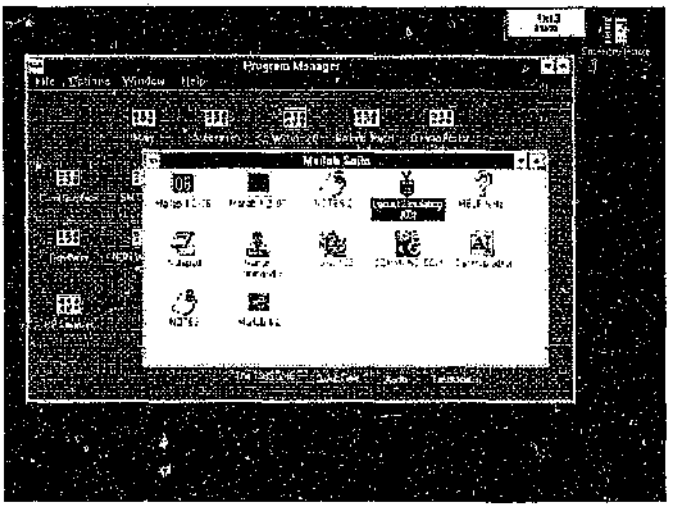


Fig. E1.1 The Windows Program Manager, showing possible icons including the Simulation Model icon.

If Matlab is correctly installed (see notes below), double-clicking the icon will result in the appearance of the Matlab Licence Information, and subsequently the simulation model's *Banner Window*. There should be no error messages displayed at this stage.

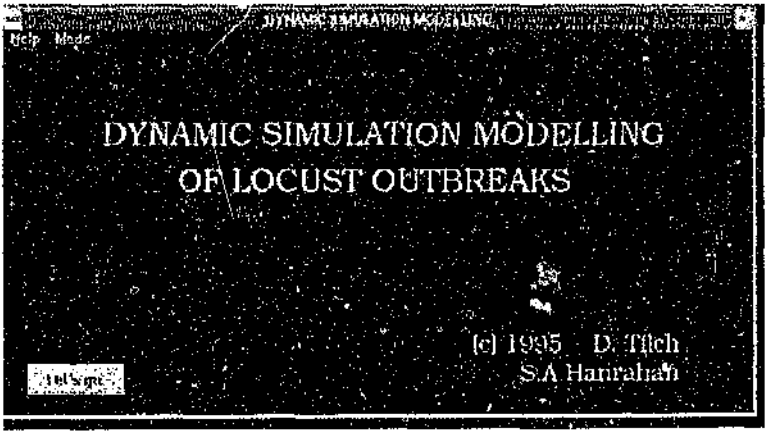


Fig. E1.2 The simulation model Banner Window.

In order to start the simulation model, left-click the mouse pointer "⇖" once on the "OK" push-button. The Banner Window will close, and will be replaced by the *Model Control Panel*. The Matlab Command Window will always be available in the background, but is not strictly required for running basic simulations.

NOTE: *In order to launch the simulation from the icon, the line CPANEL must* appear in the file STARTUP.M **that resides in your model directory.** If this file is altered or deleted you cannot launch the simulation from the icon. The icon will then only launch the Matlab Command Window, and the model will have to be invoked manually from the Command Prompt. This is accomplished by typing **cpanel** ↵ at the *Matlab Command Prompt* ">>".

NOTE: DO NOT DELETE the copy of STARTUP.M that is located with your main Matlab files. Be sure that if you decide to delete STARTUP.M for any reason, you delete only the STARTUP.M file located in the path specified as the one where your simulation model files are located.

Once the Matlab Command Window is open, all extended commands pertinent to the model are issued at the *Matlab Command Prompt* ">>" a double greater-than sign. However, the basic modelling procedures can all be invoked and executed from interactive menus and pop-up controls such as pushbuttons.

You can access the Matlab Command Window even while the model control panel is open. To enter commands into the Command Window, you need to bring it to the foreground. This is done simply by left-clicking once somewhere in that window.

If you have difficulty finding windows that have been minimised, or have become obscured behind other foreground windows, you can get a listing of all current windows, as shown in Figure A1.3. Touch the "Ctrl" key - and hold it down. While still holding down the "Ctrl" key, touch the "Esc" key. Release both keys together. You will be shown a listing of all current applications. Double left-click the mouse pointer on the name of the window you want brought to the foreground.

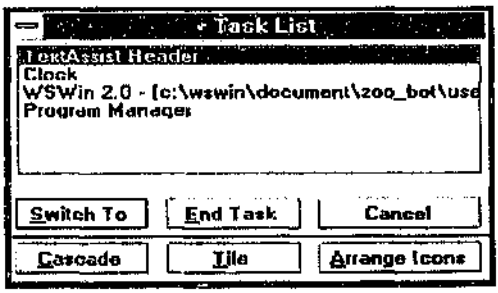


Fig. E1.3 The Windows task manager, used to select the foreground application.

If you want to verify what revision of Matlab you currently are using, type **ver** ↵ at the Matlab Command Prompt. Information similar to the following will be displayed:

```
MATLAB Version 4.2
MATLAB Site Identification Number: 374705
MATLAB Toolbox Version 4.2 25 Mar 94
SIMULINK model analysis and construction function. Version 1.3  18
Mar 94
SIMULINK demonstrations and samples.   Version 1.3  18 Mar 94
SIMULINK block library.   Version 1.3  18 Mar 94
SystemBuild 3.0 model import into SIMULINK.   Version 1.0a   09 Mar
94
```

The Command Prompt will again be displayed ">>".

NOTE: THE MATLAB COMMAND INTERPRETER IS CASE SENSITIVE. All commands should be issued in lower-case.

If you have closed the model's Control Panel, you can re-invoke the interactive mode by typing **cpanel** ↵ at the ">>" command prompt.

If you experience problems with dialogue boxes locking-up, or only partially displaying, or a mouse pointer that permanently remains as "⧖", you may need to shut down Matlab. In the Matlab Command Window, type **^c** ↵ and then **quit** ↵ - and re-starting the simulation from the icon.

QUICK OVERVIEW

Once the Control Panel is visible, basic modelling operations can be performed by selecting appropriate items (pushbuttons or menus). The details of how to perform simulations are described in the following chapter.

The dialogue boxes and windows used for driving the model are generated by Matlab script files (.M files). These are not compiled, and do not have associated executable modules. Therefore, it is an easy matter for those users familiar with Matlab's scripting language to modify and customise any of the user interface (UI) features, such as windows, menu bars and dialogue boxes, merely by editing the appropriate .M file.

Appendix E Section 2 · Running a trial simulation

LOADING WEATHER DATA FOR A SIMULATION RUN.

Before a simulation can run, three pieces of information are required:

1) Weather data for the simulation period.
The rule based model makes decisions concerning transitions between lifecycle stages based on the prevailing weather data. Numerical values for maximum soil temperature, minimum soil temperature, maximum air temperature, minimum air temperature, rainfall, wind speed, wind direction, are required for every day of the simulation.

2) The specific sites involved.
Migrations of populations from one site to any other are based on the interaction of the daily wind data, and the exact geophysical relationship between the sites. Co-ordinate data locating the sites is necessary to determine the bearing of any site relative to another, and the distances between sites.

3) Initial locust population conditions.
The initial populations of all lifecycle stages (all stages of eggs, hoppers, maturing adults, flying adults) are required so that the simulation can progress from some known starting point. Logically, these values would be the ending conditions from a simulation that had been run to cover a time period immediately before the one currently of interest.
If no simulations have been done to estimate these initial conditions, then values derived from field observations should be used.

All the required information is already available for certain sites, making it possible to be up-and-running immediately. To perform simulations for additional sites, or for time periods not already provided for, you will have to provide data files in the necessary format (See the section E6 - 'Adding your own weather data').

After choosing the "OK" button from the Banner Window, you will see the Control Panel dialogue box.

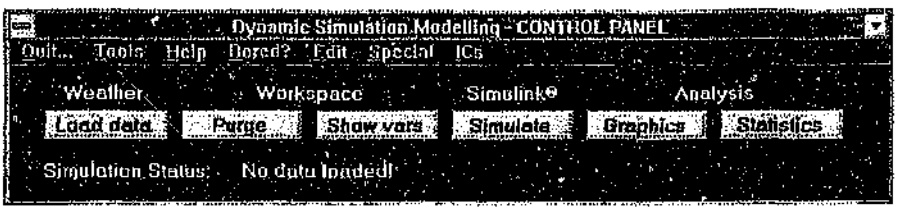


Fig. E2.1 Control panel dialogue box for the simulation modelling software.

Choose the "LOAD DATA" push-button. The loading dialogue will appear.

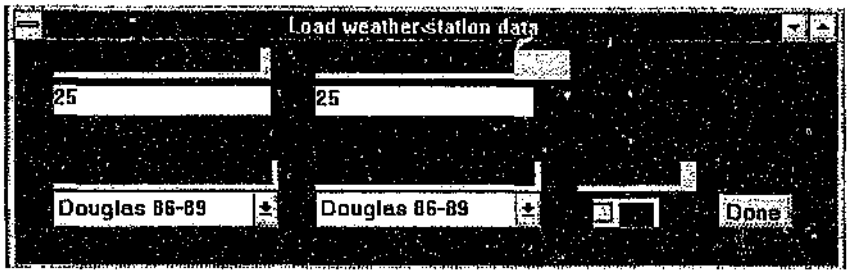


Fig. E2.2 The data loading dialogue box.

You need to select which sites are to be used for the simulation, as well as whether you want migrations to be simulated or not. You may **not** alter the flight threshold temperature, and the degree-day threshold for development, even though these values are displayed here. These values must be altered in the PRECALC.M script files if they are to take effect. These two text entry boxes are included here only to be implemented in a future revision of the model.

Two site-name fields are displayed in the dialogue window.

To choose site A for a simulation, click the "⇩" push-button to expand the drop-down menu. A list of town names will be displayed. Select the town of your choice for station A, by left-clicking the mouse pointer on that name. In the same manner, click the "⇩" push-button to expand the drop-down menu for station B site selection. A list of town names will be displayed. Select the town of your choice for station B, by left-clicking the mouse pointer on that name.

NOTE: DO NOT ENTER THE SAME SITE for two different stations. This will result in a nonsensical simulation. Error trapping built into the simulation model will warn you of this condition and require you to enter an alternative site for A or B.

Next, look at the checkbox for migration on/off:

```
☐ migration ON/OFF
```

At this stage you can select whether the simulation will allow populations to move between sites if the weather is conducive to migration. Or you can inhibit migrations even when weather conditions indicate that it should occur. It is possible at any subsequent stage to turn migration ON or OFF, regardless of what you have selected here.
Left-click the mouse pointer in the little checkbox , to select migration on "☒" or off "☐". Repeated clicking in the box toggles migration on, off, on, off, etc.

Once you are happy with the chosen sites, and the other options of this dialogue window, left-click on the "OK" push-button.

The simulation responds with a dialogue window:

```
LOADING DATA FILE...
Loading weather data for station A
```

If your file format is ASCII text, then follows a display in the Command Window of the entire weather data file for the station chosen to be station A:

```
  1  356  104  388  240   0   1   1  30  50  30  230
...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...
...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...
365  388  155  415  263   0   1   1   0   0  80  230
```

If you are using Lotus™ 1-2-3 worksheets as data files, there is no such data display.

The correct reading of data is indicated by a "Done." message, and confirmation that station A is in fact associated with a named town.

```
Done.

Currently working station A as Douglas
```

A sequence of preliminary data calculations are performed at this time. Three large matrices called Wetindex, DegreeDay-Air and DegreeDay-Ground are created. Correct completion of these calculations is indicated by the" Done." message:

```
CREATING wet-index DATA STRUCTURES...
Done.

CREATING degree-day VARIABLE STRUCTURES...
Done.
```

The initial conditions for the first station are now loaded. These data refer to how many individuals there are in each egg and lifecycle stage. Again, the "Done." message indicates a correct reading of the data file.

```
LOADING station A initial conditions...
Done.
```

The same calculations are now repeated for each and every other site / station to be simulated. Sites are named sequentially "A", "B", "C", and so on.

The dialogue box shows:

```
LOADING DATA FILE...
Loading weather data for station B
```

and in the Command Window will - if you are using ASCII data files - be displ[illegible] the data:

```
LOADING weather data FILE for station B...

    1  356  104  388  240   0   1   1  30   50   30  230
  ...  ...  ...  ...  ...  ...  ...  ... ...  ...  ...  ...
  ...  ...  ...  ...  ...  ...  ...  ... ...  ...  ...  ...
  365  388  155  415  263   0   1   1   0   0   80  230

Done.

Currently working station B as Pofadder
CREATING wet index DATA STRUCTURES...
Done.

CREATING degree day VARIABLE STRUCTURES...
Done.

LOADING station B initial conditions...
Done.
```

NOTE: If you encounter any error messages or warning messages, it is recommended that you check the structure of the data files, and the format of the commands you have issued. It may or may not be possible to proceed with a simulation - depending on the nature of the error, but it is likely that the results will be meaningless.

The simulation model will next automatically load the default initial conditions for sites in use. The initial conditions specify the state of all populations up until the start of the current simulation (day zero). Thus any simulation can be made aware of any populations that are assumed, or known, to exist prior to the date of that simulation commencement. Two simulations can be run end-to-end, generating exactly the same results as a single longer simulation, if the second simulation uses initial conditions that are equivalent to the ending conditions of the first simulation. The default initial conditions (specified by the two files ICSA.M and ISCB.M for sites A and B) may not be those you wish to use. To load other initial conditions, choose "**ICs**" from the Control Panel Menu Bar, and then select the desired initial conditions from the list.

All the simulation variables are now loaded. It is possible at this point to view and or modify any of the variables in use.

The Control Panel dialogue box remains on screen:

```
                Dynamic Simulation Modelling - CONTROL PANEL

menu bar:   [Quit · Tools · Help · Bored · Edit · Special · ICs ]

pushbuttons: [Load Data] [Purge] [Show Vars] [Simulate] [Graphics] [Statistics]

      Simulation status: Station A = Douglas, Station B =
Pofadder
```

NOTE: Before proceeding, you have to make certain variables available to the Simulink graphical companion program. This is achieved by choosing from the CONTROL PANEL menu bar:
Tools, and then **Globalise**

Then, left-click the mouse pointer on the push-button of your choice. (suggestion: **[Simulate]**)

"Simulate" launches the Simulink simulation module that models the lifecycle and the migrations.

"Show vars" displays all the variables currently in use and the memory they are occupying.

" Purge" only in exceptional circumstances. If there has been an error, and the simulation model fails to clean up any unnecessary variables, Purge clears all non-population variables.

The following two function CANNOT be used until after the simulation proper has been performed.

"Graphics" graphically displays (plots graphs) simulated results.

"Statistics" performs various statistical tests (rank correlation) on the simulated data.

Matlab Command Prompt commands useful at this stage are:

what shows the contents of the current directory (the directory in which all the simulation files are located).

who Lists the names of all the variables currently defined.

whos Lists all variables currently in use and provides information on their size, number of elements (in the case of a matrix variable) and the RAM occupied by the variable. This command is useful to determine how close you are to overtaxing your computer's resources. (Same as pressing the "Show vars" push-button)

name Typing the name of any variable will explicitly display the contents of that variable (if it is a matrix) or its value (if it is a scalar).

name= The assignment statement redefines a variable. If you wish to change the content, or value of any variable, issue a command such as: ***name = new value***↵
For example, the variable "yn" is a flag that indicates whether migration should be allowed or not. Typing **yn**↵ will yield the response "yn = n", or possibly "yn = y". To modify this variable, issue an assignment statement: **yn='y'**↵.

qui Displays the long (full town names) of the sites currently loaded into the simulation.

You can choose to access the Command Prompt by left-clicking anywhere in the Matlab Command Window; you can return to the Simulation Option dialogue by left-clicking in the Simulation Run Options window.

You may also load any set or sets of initial conditions that you have created, simply by typing the name of the M file defining those initial conditions. Should you wish to create your own set of initial conditions, it is suggested that you look at the files ICSA.M and ICSB.M to learn the format of such files. Create your own initial condition file with exactly the same structure, give it a suitable name, and edit into it the appropriate values for each population size. Those populations can be used as initial conditions by typing the file name at the command prompt.

It is possible to add your custom initial conditions file to the Control Panel menu bar, but this requires editing the CPANEL.M script, and it is not suggested for beginners or those unfamiliar with the Matlab scripting language.

COMMENCING A SIMULATION RUN

NOTE: You cannot commence a simulation run before loading weather data for at least two sites - see the section on loading weather data at the beginning of Section E2.

To commence a run, you need to start Simulink™. This is accomplished by choosing the "Simulate" push-button from the Control Panel.

Simulink will start and automatically load the *Simulation Block Diagram*. The Block Diagram appears in a newly opened window that will overlay the Matlab Command Window, and the Control Panel. The block diagram in the Simulink Eggcyc09 window is shown as in Figure E2.3, overleaf.

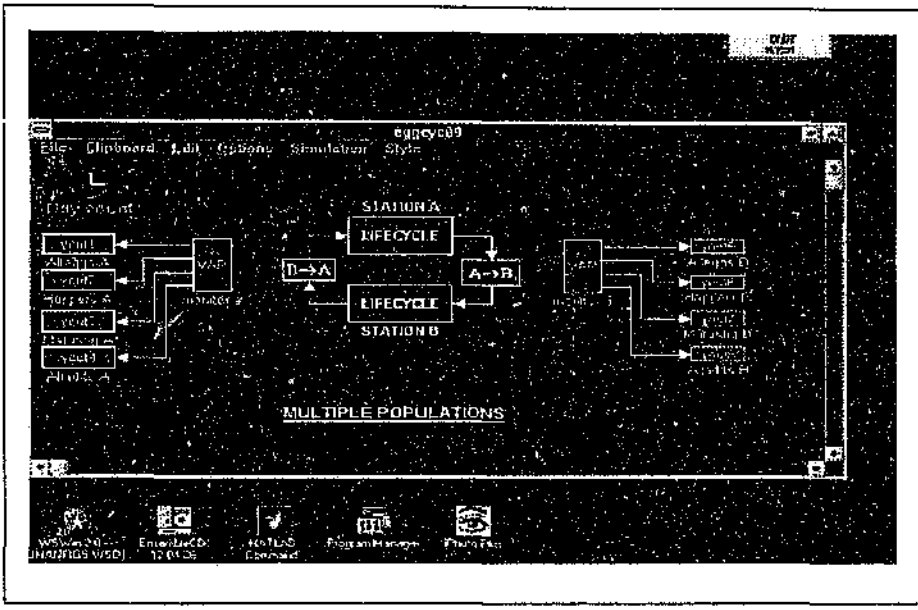


Figure E2.3: Simulation Block Diagram

It is possible to flip any window to the foreground merely by left-clicking the mouse cursor once somewhere on that window.

The Block Diagram window shows a graphical representation of the simulation model. Initially only a two site model is provided. You may wish to expand this in future to a three, or four, or more site model.

The *Menu Bar* associated with the Block Diagram Window provides a number of options.

File	Clipboard	Edit	Options	Simulation	Style

Each menu command has an underscored letter in the name. These menu items can be accessed very quickly by pressing the Alt key and simultaneously the appropriate underscored letter, instead of using the mouse. This applies to all the menus you have seen and used up to this point also!

Left-click once on the word **Simulation**, or simply use Alt s.
A drop-down menu appears....

Start	Ctrl+T
Resume	
Continue	
Parameters	

Left-click the mouse cursor once on the word **Start**. Alternatively, Hold down the "CTRL" key and simultaneously touch the "t" key.

MONITORING A SIMULATION'S PROGRESS

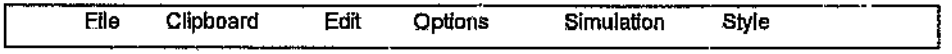
The simulation has now started. If you wish to view the progress of the simulation, double left-click the mouse cursor on the clock icon "⏲" within the Block Diagram Window. An additional window will open and display the current day of the simulation. This figure will increment at a rate of about two days per second, depending on the capabilities of your computer, and the number of sites in use.

When the simulation is complete, the bell will sound.

NOTE: The bell will also sound if there has been an error during the simulation. If an error has occurred, an Error Window will open in the foreground containing an error message.
Further details of the error may be displayed in the Matlab Command Window, which will have to be brought to the foreground for viewing.

ADJUSTING THE DURATION OF THE SIMULATION

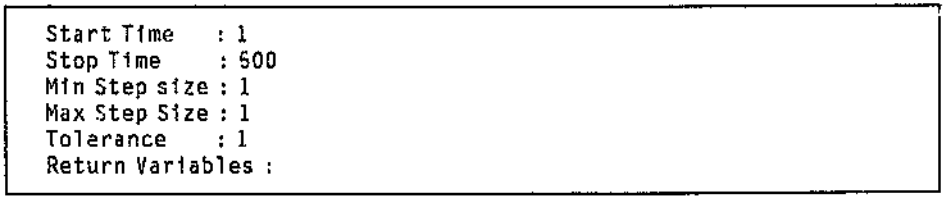
The simulation runs for a duration of *N* days. *N* is the value contained in the *stop time* field of the *Simulink Control Panel*. The Simulink Control Panel can be accessed from the Block Diagram Window's Menu Bar.



Double left-click on the Simulation word, (or use Alt s).
A drop-down menu will be displayed...



Select the Parameters option by double left-clicking it, or use Alt m. The following fields will be displayed....



Place the text cursor in the Stop Time field, by single left-clicking the mouse cursor somewhere in this field. Use the "Backspace" key or "Del" key to remove the numeric value and type in a new value.

Left-click on the "OK" push-button to close the Simulink Control Panel window.

NOTE: Do not alter the values contained in the fields *Min Step Size*, *Max Step Size*, *Tolerance*. These values are used to determine the step sizes used when iterative calculations are performed, and the simulation step size.

NOTE: Before changing the duration of the simulation, be aware of the size of the available data file. Do not set the *Stop Time* of the simulation to a value greater than the number of days in the weather data file. If you do so, the simulation will run out of weather data prematurely, and terminate unexpectedly.

NOTE: You can set the *Start Time* of a simulation to a value other than "1". This might be desired if you wish to start a simulation from a date that is sometime after the first date referenced in the weather data file. For example, the weather data file may contain data from Sept 01, 1985 through Aug 31, 1995. However, you may wish to simulate from the beginning of 1986. To do so, set the Start Time = 122, since January 01, 1986 is 122 days after Sept 01.

NOTE: The *Stop Time* is NOT the duration of the simulation. The duration (in days) is the difference between the *Start Time* and the *Stop Time*.

RETRIEVING THE RESULTS OF THE SIMULATION

As a simulation progresses, the variables associated with each lifecycle stage are stored. This record is updated for each day of the simulation. Thus at the end of a simulation run there is available a record of the numbers in each lifecycle stage, spanning the entire simulation interval. Experienced users may wish to access and plot these variables directly. The section E3 - 'Customising The Simulation Output' - explains how to do this.

Should you wish to view the results quickly and not to retrieve variables manually, then you can make use of one of the modules provided for this purpose. These modules are simply Matlab script macros. They can be customised if desired - see section - 'Customising The Simulation Output'.

Available output scripts:	**show**	Display 4 graphs on screen, of simulated results and of actual recorded outbreak numbers, for both stations A, and B. Simulated results show hoppers, maturing adults, mature adults.
	dumpWK1	Dumps the simulation results to disk. Uses interactive dialogues to get and check variable names and file names. Stores the file in Lotus 1-2-3 .WK1. format for direct use in most spreadsheets.
	logplot	Plots the numbers of hoppers, maturing adults, and adults on log-linear axes, to facilitate easy discernment of small populations
	eggplot	Plots the egg reserves (egg stages defined by the monitor scripts) on linear axes.
	rain	Plots the rainfall curves for the two sites in use.

These scripts are available from the **Special** option of the Control Panel menu bar. The Control Panel may need to be brought to the foreground from behind the Simulink Block Diagram window.

For example, to see the plot of hoppers, maturing adults, and mature adults for two sites, as well as the recorded outbreak numbers select **Scripts** from the Control Panel menu bar, and then select **Show**.

Several output scripts can be invoked sequentially, in any order. You may wish to close down those display windows that you are no longer interested in, to conserve computer resources and to reduce clutter on the screen.

NOTE: These output scripts rely on the output vector of the script files MONITORA and MONITORB. If either of these two script files is changed, the results plotted by any or all of the above menu choices may be unpredictable.

NOTE: You may see additional entries in the SPECIAL section of the menu bar that are not outlined here. These are scripts used for testing and evaluation of the model during development.

EXECUTING A SECOND SIMULATION RUN

Once a simulation run is complete it is possible to run a subsequent simulation immediately. In general this should only be required if you have chosen to split the interval of interest into several sub-intervals. For example, you may wish to simulate from Sept 01, 1985 through Aug 31, 1991. In order that the resolution of the graphs be retained you might have decided to perform this simulation in three distinct runs, namely :

Run 1 : Sept 01, 1985....Aug 31, 1987
Run 2 : Sept 01, 1987....Aug 31, 1989
Run 3 : Sept 01, 1989....Aug 31, 1991

NOTE: A second simulation run - using the same weather data as the preceding run - will yield unexpected results unless you reset the initial conditions. The second simulation run regards its initial conditions as the ending conditions of the preceding simulation run

If you choose to perform several simulation runs on the same weather data, a special approach should be used. This situation may arise if you decide to perform many simulation runs, but choose to alter some of the lifecycle or other model parameters between runs. Before starting the second run, you will need to re-load the initial conditions, otherwise the simulation runs as if it were one long run - continually building on previous population tallies.

To reset the initial conditions:

From the **Tools** option in the Control Panel menu bar, select **Reset ICs**

Alternatively, in the Matlab Command Window, type **icsa** ↵ to reset initial conditions for station A, type **icsb** ↵ to reset initial conditions for station B, and so on. Be certain to reset initial conditions for all stations in use (A, B, C, D etc.).

UNLESS you have some specific reason for retaining the ending conditions for a particular station. This method allows you to selectively reset initial conditions for some stations and not others.

You may wish to load a special set of initial conditions that you have created. In this case, enter the name of the file referencing your initial conditions at the Matlab Command Window Prompt ">>".

A number of sets of initial conditions suitable for the provided weather data are included. These may be loaded by choosing the appropriate entry from the Control Panel menu bar option "ICs". After choosing "ICs", a list of available initial conditions will be displayed. Choose the one(s) applicable to the sites and simulation durations you are using.

Matlab will respond to each command with an appropriate message such as...

```
LOADING station A initial conditions...
Done.
```

You can also commence your second or subsequent simulation runs by choosing the **Load Data** pushbotton in the Contol Panel, in the same way you did to start the first simulation. This will ensure that all initial conditions are reset. However, it has the disadvantage of being much slower since all the necessary weather data will be reloaded from disk files.

To find out more on how to change and customise the initial conditions, see Appendix E section 4 - 'Customising The Simulation Input'.

Appendix E Section 3 Customising the simulation output

HOW POPULATION RECORDS ARE STORED

Each lifecycle stage is represented internally by a single dynamic data structure. This is essentially a number, or an organised set of numbers that reflects how many individuals are present at a particular simulation day. Each of these numbers is continually changing in size, from day to day, since they only reflect the population sizes on the current day of the simulation.

In order to recover a meaningful result from a simulation it is necessary to record the history of every one of these population data structures. In other words, it is necessary to keep a record of each of their values on every day of the simulation. This operation is performed by a script file called generically MONITOR_.M There is a separate MONITOR script for every station in use. Hence, a two site simulation model will have two monitor scripts MONITORA.M and MONITORB.M for station A and station B.

Within the MONITOR_.M script file are the instructions to concatenate the various fields that comprise each life-cycle stage's variable. This is necessary since some of the life-cycle stage variables are not merely numbers, but are two-dimensional arrays whose entries need to be tallied to get a population figure. The lifecycle uses the following internal variables, where the underscore can be either A or B or C etc., depending on which station the script file applies to.

RAINFALL_	MATURINGADULT_	LAY1_	LAY2_
LAY3_	LAY4_	ADULTMALES_	INSTARS_
DIAPAUSE_	DIAPAUSE1_	DIAPAUSE2_	TURGID1_
TURGID2_	QUIESCENT1_	QUIESCENT2_	
NONDIAPAUSE_			

Because of the structure of these variables it is not valid to assume that any variable named above simply represents a daily population figure. Incorporated with the population into the data structure are certain binary flags indicating the past history of those individuals, and sometimes also accumulated developmental degree-days.

MONITOR_ extracts the values from each of these lifecycle variables on a daily basis and makes the total daily population for each stage available as...

diap	diap1	diap2	q1	q2	turg1
turg2	instars	nondiapmat	nlay1	nlay2	nlay3
nlay4	adults				

These variables are direct numerical tallies of the number of individuals on the current day of simulation. However, they are not global variables, meaning that they are only available to instructions within the MONITOR_ script. No other script or function or procedure has access to these values as calculated here.

MONITOR_ has access to the following variables, and any of them, or combination of any of them are available as the numerical output from the simulation:

diap	Number of Diapause eggs
diap1	Number of Diapause1 eggs
diap2	Number of Diapause 2 eggs
nondiap	Number of NonDiapause eggs
q1	Number of Quiescent 1 eggs
q2	Number of Quiescent 2 eggs
turg1	Number of Turgid 1 eggs
turg2	Number of Turgid 2 eggs
instars	Number of hopper in instar [i] through instar [v]
mat	Number of maturing adults
nlay1	Number of sexually mature adult females in Lay1 stage
nlay2	Number of sexually mature adult females in Lay2 stage
nlay3	Number of sexually mature adult females in Lay3 stage
nlay4	Number of sexually mature adult females in Lay4 stage
adults	Number of sexually mature males & females in all laying classes

HOW INTERNAL POPULATION VARIABLES BECOME THE OUTPUT

The MONITOR_ scripts generates FOUR output channels per station. Thus you could, for example, keep track of nlay1 and nlay2 and nlay3 and nlay4 individually. This would occupy all the output channels , so you could not keep track of any eggs, or hoppers.

However, you could combine (nlay1 + nlay2 + nlay3 + nlay4) so that they would together only take a single output channel. This would leave three remaining channels through which to keep track of eggs. Note that now, by combining, you can no longer keep track of each of nlay1, nlay2, nlay3, nlay4 INDIVIDUALLY.

It is possible to increase the number of output channels of the MONITOR_ script. This requires a thorough understanding of Simulink and how it is controlled by the Matlab interpretive language. A value of four was selected as a compromise between quantity of information recoverable and the amount of storage space required. Adding more channels makes more demands on the computer's available memory.

Redefining the output channels of the MONITOR_ script requires editing the MONITOR_.M script file with a text editor such as Norton DeskedIt, DOS's EDIT, Norton Commander EDIT, Windows NOTEPAD, or a wordprocessor that is capable of saving text in unformatted ASCII.

HOW TO EDIT THE MONITOR SCRIPTS

Use the editor of your choice to open the MONITOR_.M script file that you wish to modify.

NOTE: The scripts available from the **Special** option of the Control Panel menu bar make implicit assumptions about the structure of the output vector of the monitor scripts. If the output of the monitor scripts is changed, the scripts **logplot**, **eggplot**, **dumpWK1** etc. will not necessarily return the expected results.

NOTE: Remember that the names of the MONITOR_ scripts do NOT contain an underscore as denoted here. The underscore "_" will be replaced in the filename by a letter "A", "B", "C", etc., to denote the station that the script monitors. Specifically, the monitor script for station A will be designated MONITORA.M. Similarly the monitor script for station B will be designated MONITORB.M.

Once the MONITOR_ script is open, scroll down to the last line. This should be of the form...

`output = [ component_one component_two component_three component_four];`

NOTE: The punctuation and parentheses are important!

The fields *component_one component_two component_three component_fr* can be replaced by any of the following variables:

diap, diap1, diap2, nondiap, q1, q2, turg1, turg2, instars, mat, nlay1, nlay2, nlay3, nlay4,adults

or any algebraic combination of any from the list. The meaning of each of these variables is explained : .

For example, changing the output assignment to:

```
output          = [diap1+diap+diap+q1+q2+turg1+turg2 instars mat
adults];
```

results in:

- output channel one being the tally of all eggs that are not non-diapause eggs,
- output channel two being the tally of all instar stages,
- output channel three being the tally of maturing adult stages,
- output channel four being the total of all adults, male and female that are sexually mature.

You may wish to introduce a new variable that represents a sub-tally, which will simplify your output statement. In this example a new variable is introduced - called "alleggs". Notice that before alleggs can be used in the output statement, it has to be defined in its own assignment statement. This requires that you add the additional `alleggs =` assignment statement

```
alleggs     = diap1+diap+diap+q1+q2+turg1+turg2+nondiap;
output      = [ alleggs instars mat adults];
```

NOTE: You may wish to retain the original last line of the MONITOR_ script so that you can always go back to using it as it was. There is no need to delete the existing `output =` assignment statement in the last line, simply make sure that your new line is below any other such assignments. The only assignment that will actually define the output is the last one in the file.

Similarly, if you define a new variable, and subsequently wish for a different definition of that variable, you can retain the older definition for backup purposes. Be sure, however, that the assignment that you want to be the effective one is the last one in the file. For example, if your script file looks like this...

```
alleggs    = q2+q1+nondiap;
alleggs    = diap1+diap+diap+q1+q2+turg1+turg2+nondiap;
output     = [ alleggs diap1 diap2 mat];
output     = [ alleggs instars mat adults];
```

the only lines that define the output are the last ones of each type:

`alleggs = diap1+diap+diap+q1+q2+turg1+turg2+nondiap;` and
`output = [ alleggs instars mat adults];`

Alternatively, you can precede any line that you want the interpreter to ignore, with an "at" symbol "@". In this case, the order of the 'commented out' instructions is irrelevant. Repetitive non-commented instructions will still always be interpreted in their order of occurrence.

Once you are satisfied with the modifications you have made, be sure to save the updated file, using whatever commands your editor requires.

Windows editors such as Norton Deskedit, and Notepad, as well as DOS's EDIT all use "ALT f s" as a file-save keystroke sequence.

NOTE: When using any multitasking operating system, such as Windows, you may get an error:
`ERROR: SHARING VIOLATION ON DRIVE C:`
if you try to edit and then save a file that is currently in use by another application (in this case Matlab's Command Interpreter).
Should this happen, you will need to free the file you wish to edit from the control of the other application, by shutting down that second application. Only then will you be able to save the file successfully. This problem can be avoided by enabling the operating system's file sharing system. Under DOS/Windows 3.1 this is done by loading SHARE.EXE in the CONFIG.SYS. See your DOS manual for instructions on this procedure.

NOTE: Problems are known to occur when using the NOTEPAD application with Windows 95. Notepad, under Windows 95, does not recognise the Matlab script files as text files. Difficulties occur when trying to open the M files, and also when Matlab's interpreter tries to access the subsequently saved files.

When you next run a simulation, the four output variables available for plotting etc., will be those that you have named explicitly in
`output = [ component_one component_two component_three component_four];`

NOTE: If you want the output variables of all stations in use to be changed similarly, you will need to edit all the associated MONITOR_.M script files, namely MONITORA.M, MONITORB.M, MONITORC.M, etc.

NOTE: The variable names you use in place of *component_one, component_two, component_three, component_four* are NOT AVAILABLE at the Matlab Command Prompt AS TALLIES OVER THE WHOLE SIMULATION PERIOD. They are available at the command prompt only as populations on a single day. The tallied populations are stored as a n array, or list (one entry for each simulation day). The corresponding list names are:

STATION A

output component name	**listed variable name**
component_one	yout1
component_two	yout2
component_three	yout3
component_four	yout4

STATION B

output component name	**listed variable name**
component_one	yout5
component_two	yout6
component_three	yout7
component_four	yout8

STATION C (if it exists)

output component name	**listed variable name**
component_one	yout9
component_two	yout10
component_three	yout11
component_four	yout12

It is these listed variables (yout1, yout2, etc.) that should be used to plot the outputs, or viewed to determine population histories.

It is important to keep track of which internal population variable (diap, t2, q1, etc.) is mapped to which listed output variable (yout1, yout2, etc.). This is especially important if you wish to modify any of the plotting routines.

HOW TO USE THE OUTPUT LISTS - SIMPLE OUTPUT DISPLAYS

The output lists yout1, yout2, etc., exist in groups of fours, because each station, or site, generates four output lists . So a two-site model will have eight output lists yout1...yout8. A three -site model will have twelve output lists yout1...yout12, and so on.

These lists are the most useful available output record, since they each contain the variable-of-interest 's history over the entire simulation.



Any day of the simulation can be pinpointed in any list, if you know the index of that day. The index is simply a number that tells you which day of the simulation it points to. For instance, the ninety-seventh day of the simulation would have a day index of "97". The first day would have an index of "1".

For example, if you wanted to see what the population of mature adults on the 97th day of the simulation was, you would proceed as follows:

Make certain that **adults** appears somewhere in the
`output=[ component_one component_two component_three component_four];`
statement of the MONITOR_.M output script. Suppose the output assignment statement was...

```
output = [ alleggs instars mat adults];
```

Note carefully the position of the **adults** variable - it is the **fourth** in the assignment.

If **adults** does not appear in the `output =` assignment, you will have to edit the MONITOR_ script to include **adults**, or whichever population variable you have chosen to look at. Remember, you need to change the MONITOR_.M script for each station if you wan to monitor **adults** for all the stations.

Since **adults** is the fourth population variable in the output assignment, it will be recorded in yout4 if we are concerned with station A, it will be recorded in yout8 if we are considering station B, and it will be recorded in yout12 if we are considering station C, and so on.

Generating a plot of yout4 would thus give you a plot of mature adults at station A over the whole simulation interval. A plot of yout8 would show the population of all mature adults at station B for the entire simulation interval.

To find the number of mature adults on day 97 at station A, issue the following command at the Command Prompt: **yout4** ↵
The number will be displayed in the Command Window.

A simple plot can be obtained by issuing the *plot* command in the Command Window. At the Command Prompt, type **plot yout4** ↵

Matlab will respond by opening a *Figure Window*, in which it will display a single graph of the mature adults, plotted over the entire simulation period.

MANIPULATING PLOT VISUAL PARAMETERS

Some useful commands to manipulate the plot obtained above are:

axis([x_{MIN}, x_{MAX}, y_{MIN}, y_{MAX}]) ↵ sets the limits of the plotted axes - allows details to be shown

title(*name*) ↵ names a plot

UNDERSTANDING THE "SHOW" SCRIPT

For those users wishing to obtain a plot directly, and not wishing to design their own plots, some simple scripts are included to generate graphical displays of the simulation results.

show.m can be invoked once a simulation run has been completed. From the Control Panel menu bar, select **special**, and then **show.**

A Figure Window will be opened and brought to the foreground, and four sets of axes will be displayed. These show clockwise from top left: simulated resulted for station A, simulated results for station B, actual outbreak records for station B, actual outbreak records for station A.

SHOW.M is a script file containing a list of more complex commands. In this sense it does no more than can be accomplished by issuing these commands from the Command Prompt. However, concatenating these commands into a single script provides a macro command, saving you the effort of having to issue all the necessary commands.

WHAT SHOW.M ACTUALLY DOES

Below is a listing of the SHOW.M script. After each line is an indented explanation of that lines function. In this way you can see exactly how a complex display is built up.

```
t=[1:length(yout2)];
```
establishes the length (number of days) of the simulation
```
subplot(2,2,1),bar(t,yout4,'b');
```
plots a one-quarter size plot of yout4, in blue ink, in the top left hand corner
```
hold on;
```
freezes the plot already done so new plots can be overlaid on existing plots
```
subplot(2,2,1),bar(t,yout2,'g');
```
plots a one-quarter size plot of yout2, in green ink, in the bottom left hand corner
```
axis([100,500,0,max([max(yout2) max(yout4)])]);
```
sets the x-axis to show day 100 to day 500. Scales the y-axis to include the largest value
```
subplot(2,2,3),plot(LCOUNTa),'w';
```
plots in the bottom left corner, in white ink, actual number counts for station A
```
title(namea);
```
assigns the name of stdation A to the left-hand side plots
```
axis([100,500,0,max(max(LCOUNTa))]);
```
sets the x-axis to show day 100 to day 500. Scales the y-axis to include the largest value
```
subplot(2,2,2),bar(t,yout8,'b');
```
plots a one-quarter size plot of yout8, in blue ink, in the top right hand corner
```
hold on
```
freezes the plot already done so new plots can be overlaid on existing plots

```
subplot(2,2,2),bar(t,yout6,'g');
```
plots a one-quarter size plot of yout6, in green ink, in the bottom right hand corner
```
axis([100,500,0,max([max(yout6) max(yout8)])]);
```
sets the x-axis to show day 100 to day 500. Scales the y-axis to include the largest value
```
subplot(2,2,4),plot(LCOUNTb),'w';
```
plots in the bottom right corner, in white ink, actual number counts for station B
```
title(nameb);
```
assigns the name of stdation B to the right-hand side plots
```
axis([100,500,0,max(max(LCOUNTb))]);
```
sets the x-axis to show day 100 to day 500. Scales the y-axis to include the largest value

NOTE: The copy of SHOW.M you have may not look **exactly** as shown due to constant updates and improvements:

It is possible to edit the commands in the SHOW.M script to customise it for your own use. SHOW.M can be edited with any text editor as outlined in this chapter, under the section 'HOW TO EDIT THE MONITOR SCRIPTS'.

UNDERSTANDING THE "NUTAB" SCRIPT

Also provided is a script NUTAB.M, for dumping the simulation lists to a disk file. This is useful if you wish to have a tabulated output of the numerical populations for certain lifecycle stages.

A non-interactive version of **nutab** exists. This is called **tab**. **Nutab** is easier to use, and is recommended over **tab**. Tab is invoked from the Matlab Command Prompt ">>", by typing **tab** ↵.

Before NUTAB can write the numerical populations to a file, those populations need to be assigned within the MONITOR_ script(s) - see notes below. The section 'HOW INTERNAL POPULATION VARIABLES BECOME THE OUTPUT' explains how to ensure that the population variable that you are interested in is included in the simulation output (the list variables - yout1, yout2,...)..

Once you have assigned the population variable (s) you are interested in (e.g.., nlay1, instars, q1, q2, ...) to the desired list variable (e.g.., yout1, yout2, ...), you need to perform a simulation run.

NOTE: You must assign the population variables to list variables BEFORE you perform the simulation run. List variables (yout1, yout2,...) are not updated after a simulation run has finished, hence re-assigning population variables to these lists after a run will not affect the lists.

NOTE: You do **not** need to re-assign the population variables to a desired list variable each time you perform a simulation. Once the assignments have been made they are retained for all subsequent simulations. The need for re-assignment only arises if you wish to **change** the particular mapping of population variables-to-list variables.

Perform the simulation as you would normally do. Allow the simulation to run to completion.

From the Control Panel menu bar, select **special**, and then **nutab.**

Matlab will respond with a pop-up dialogue:

```
Select list variable name to dump :
```

You should now choose the *list* variable that you want tabulated. This will be one of yout1, yout2, you3,...yout8 if you are running a two-site simulation. If you have a three-site model, the list includes yout9 up to yout12. Nutab automatically detects which list variables exist, and only displays those available.

Left mouse-click on the name of the list variable. e.g., **yout4** ↵

If you have no list variables in the workspace (either because you have not yet run a simulation, or you have not assigned any output list variables, or you have cleared any previously existing list variables), Matlab will respond with a pop-up error window:

```
.There are no list variables to tabulate!
```

and you will be returned to the Command Prompt. Correct the problem and re-issue the **nutab** command.

If the list variable is valid, the entire list will be displayed on-screen.

```
    1     80
    .      .
    .      .
    .      .
    .      .
  500  76000
```

NOTE: The first column (left-most column) is a list of day-numbers, running from "1" to the number of days you have selected as the duration of the simulation.

NOTE: The format of this display (and of the file stored to disk) is:

Column one reflects the day index. Days are numbered sequential from the first day of the simulation.
Each row represents a single day of the simulation.
Each column contains the list variable you have chosen to tabulate. Each time you run **nutab**, an new column is added on the right. This corresponds to the most recently selected list variable.

Next will be displayed:

```
This result has been stored to disk under the file SIMRESLT.TXT
                        [Close]    [Ok]
```

The file SIMRESLT.TXT is an ASCII file containing exactly the numbers you have seen on-screen. You may wish to import this file into a spreadsheet, or text editor, which will facilitate detailed analysis of the numbers. Click on either "Close" or "Ok" to close the information window.

You are also offered the choice of saving this data to a file of another name. This is useful since the file SIMRESLT.TXT will be overwritten each time to execute the **nutab** function. A pop-up window will ask if you wish to save the file as a permanent file:

```
Do you want to store this as a permanent file?
                    [Yes]     [No]
```

Left click on "yes" or "no" as appropriate.
If you select "yes", a pop-up dialogue window will appear to allow you to select or enter the file name of your choice.

NOTE: You can use any file name, but be careful not to use a file name that is reserved for Matlab function calls. These all have the extension ".M". Also, do not use the name SIMRESLT.TXT, since this file will be overwritten next time you use **nutab**.

Matlab will also show a print dialogue window:

```
Do you want to print this result on paper?
                    (yes)     (no)
```

You now have the option of sending the entire file to the printer. Be careful about doing this, since a typical simulation run is 500 days, which will require around 10 sheets of paper, depending on your printer font.

Select y or n as desired.

NOTE: The device for printing is LPT1: (first parallel port) regardless of the Matlab "printopt" string. The device can be changed to any legal DOS device (LPT1...3, COM1..4) by editing the file NUTAB.M

NOTE: This print is performed by a COPY command issued to the DOS Command Interpreter. If a problem occurs it is most likely to be caused by too little memory for loading an additional copy of COMMAND.COM

If you choose not to print, or once the print instruction has been issued, you will be returned to the Command Prompt.

Now you might wish to alter some of the life-cycle parameters, or initial conditions, (see section - 'Customising the simulation input' and section - 'Customising the model parameters') and then re-run the simulation for the same period. After you have re-run the simulation, you can append the newer simulation results to the **nutab**ulated results on file. This is done by simply invoking **nutab** at the end of the second simulation run.

If you invoke **nutab** a *second* time, you will get a warning message:

```
NOTICE: A previous simulation record exists.

Do you want to DELETE it, or ADD to it?
```

TAB has detected that there are lists being retained in memory for storage in a file. If you do not wish to retain the tabulated results from your earlier simulation runs, choose **del**, to delete the earlier list records. If you wish to append the most recent results to all previous results, choose **add**.

NOTE: Each time you invoke **nutab**, the temporary disk file in which data has been stored is erased! Nutab will then write the current tab information to the file on disk. So, if you perform 25 simulations sequentially during a work period at your computer, you can add the results sequentially to the SIMRESLT.TXT file, and all 25 results will be retained.
If you now shut down Matlab, and restart it to do further simulations, any tabulated results gathered during the second session will overwrite and destroy the results from your earlier session. If you wish to retain these earlier 25 results you must save the results to a permanent file before running **nutab** at any time in your second session.

NOTE: If at any time you choose **del**, to delete the tab record, the previous tab results are deleted entirely. THERE IS NO UNDO COMMAND! However, the tab results will be retained in the file SIMRESLT.TXT. You should - if you have erroneously chosen to delete - immediately abort the tab command (by pressing ^C) and rename the file SIMRESLT.TXT to something else, to prevent permanent loss of that data.

UNDERSTANDING THE "DUMP WK1" SCRIPT

Also provided is a script DUMPWK1, for dumping the simulation lists to a spreadsheet file. This is useful if you wish to use a spreadsheet or data base application to analyse the simulation results..

Before DUMP WK1 can write the numerical populations to a file, those populations need to be assigned within the MONITOR_ script(s) - see notes below. The section 'HOW INTERNAL POPULATION VARIABLES BECOME THE OUTPUT' explains how to ensure that the population variable that you are interested in is included in the simulation output (the list variables - yout1, yout2,...)..

Once you have assigned the population variable (s) you are interested in (e.g.., nlay1, instars, q1, q2, ...) to the desired list variable (e.g.., yout1, yout2, ...), you need to perform a simulation run.

NOTE: You must assign the population variables to list variables BEFORE you perform the simulation run. List variables (yout1, yout2,...) are not updated after a simulation run has finished, hence re-assigning population variables to these lists after a run will not affect the lists.

NOTE: You do **not** need to re-assign the population variables to a desired list variable each time you perform a simulation. Once the assignments have been made they are retained for all subsequent simulations. The need for re-assignment only arises if you wish to **change** the particular mapping of population variables-to-list variables.

Perform the simulation as you would normally do. Allow the simulation to run to completion.

From the Control Panel menu bar, select **special**, and then **dump WK1.**

Matlab will respond with a pop-up dialogue:

```
Enter the name of the file to dump: [*.WK1]
```

The file extension is forced to be .WK1, in accordance with the Lotus 1-2-3 convention for that file format. Since the file dump is a DOS-handled procedure, only 8 characters are permitted for the file name.

Click "OK" when you have entered a suitable file name.

NOTE: There is no checking to see if a file of that name already exists! If a file by that name does exist, it will be overwritten, so be sure that you select a new name if you wish to retain previously dumped data.

The script will create the file which consists of eight columns, and as many rows as there are days in the simulation. The list variables - defined by the scripts MONITORA.M and MONITORB.M are written columnwise as follows:

(for a simulation of duration m days)

	Variables pertinent to site A				**Variables pertinent to site B**			
	yout1	**yout2**	**yout3**	**yout4**	**yout5**	**yout6**	**yout7**	**yout8**
day 1	nnn	nnn	nnn	nnn	nnn	nnn	nnn	nnn
day 2	nnn	nnn	nnn	nnn	nnn	nnn	nnn	nnn
day 3	nnn	nnn	nnn	nnn	nnn	nnn	nnn	nnn
day 4	nnn	nnn	nnn	nnn	nnn	nnn	nnn	nnn
day m	nnn	nnn	nnn	nnn	nnn	nnn	nnn	nnn

NOTE: This file is in Lotus 1-2-3 WK1 format. It cannot be viewed with a text editor or (most) wordprocessors. It can be opened directly by most spreadsheets (Quattro-Pro, Excel, etc.).

THE "RAIN" SCRIPT

The rain script plots the rainfall records for site A and site B, for the entire duration of the simulation. This is useful for evaluating whether any locust activity is associated with a particular rainfall event.

The window is resizeable to allow it and any other simulation output plots to be viewed in a tiled fashion.

Appendix E Section 4 Customising the simulation input

THE MODEL INPUT

The model input has two components:

① The simulation model requires weather data:

Daily measurements of:
- maximum air temperature
- minimum air temperature
- maximum soil temperature
- minimum soil temperature
- rainfall
- actual locust swarm counts *
- actual hopper-band counts *
- wind speed
- wind heading

* these data are not necessary for the simulation but are used by some of the plotting routines. Values of zero for each day will not affect the simulation.

The above data is required for a minimum of two sites (stations/towns). It is stored in ASCII files, or Lotus 1-2-3 worksheets, on disk, and generally should not require modification by the user.

Appendix E, section 6 "Adding your own / updated weather data" outlines the exact structure of these files and how they can be modified.

②: The simulation also requires initial population conditions:

Values representing the numbers of individuals in every lifecycle stage at the start of the simulation. These values are only required for a single day (i.e., simulation day "0"). The simulation uses these numbers as the starting points for all population calculations. If zeros are entered as initial conditions for both stations, the simulation will be static - there will be no population changes during the simulation period.

Each station has associated with it, a unique set of initial conditions. These can be altered independently for any station, without affecting any other station. The numeric values are stored in ASCII files, one for each station. Initial conditions for Station A are stored in the file ICSA.M Initial conditions for Station B are stored in the file ICSB.M, and so on.

Each ICS_.M file contains a list of assignment statements. Each statement sets a population variable (symbols that can take on any numeric value equal to the number of your choice; frequently have word-type names to help identify their function).

NOTE: Lifecycle population variables are not simple scalar values. Frequently they carry information concerning the weather history, and current developmental state of the individuals.. Hence the population variables are in many cases two dimensional arrays, or matrices. It is important to understand the function of the number in each array cell before changing it.

THE POPULATION VARIABLES

These are variables that contain the quantity of individuals at each stage of development in the lifecycle.

The population variables in use are listed below. Note that the underscore "_" takes on a value of a, b, c,... etc. to denote the site with which that variable is associated. Population variables are, for example: TURGID1_, DIAPAUSE_, LAY1_. In other words, variables for populations at station A will have the names: TURGID1A, DIAPAUSEA, LAY1A, etc. Variables for populations at station B will have names: TURGID1B, DIAPAUSEB, LAY1B, etc.

LIST OF POPULATION VARIABLES:

ADULTMALES_
LAY1_
LAY2_
LAY3_
LAY4_
DIAPAUSE_
DIAPAUSE1_
DIAPAUSE2_
NONDIAPAUSE_
TURGID1_
TURGID2_
QUIESCENT1_
QUIESCENT2_
INSTARS_
MATURINGADULT_

CUSTOMISING THE INITIAL CONDITIONS OF A SIMULATION RUN

As an example, consider the population variable DIAPAUSEA. This represents the number of diapause eggs to be found at station A. For a simulation run to take adequate account of the history of a population, DIAPAUSEA (and all other population variables) has to take on a numeric value corresponding to the diapause egg population size on the day the simulation begins. References to this variable's initial value will be found in the file ICSA.M

To modify the initial conditions, it is necessary to open the file ICS_.M using a text (or ASCII) editor.

Consult Appendix G to learn the precise structure of each population variable. For DIAPAUSE_, Appendix G provides the following information:

```
DIAPAUSE:

A 2xn matrix

1,1  1,2  1,3  1,4  ...  1,n
2,1  2,2  2,3  2,4  ...  2,n

Entries in the j-th column have been in DIAPAUSE for j days.

(1,j) = quantity of eggs laid in wet conditions, in DIAPAUSE for j
days
(2,j) = quantity of eggs laid in dry conditions, in DIAPAUSE for j
days
```

DIAPAUSE is apparently a 2-by-n matrix. This means it is a rectangular array of numbers: there are two rows, and *n* columns. (matrices are always denoted rows first, columns second, hence a *pxq* matrix has *p* rows, and *q* columns)

A diagrammatic view of the variable is shown. Each entry is denoted by two numbers separated by a comma. The first of these numbers denotes the row that the entry belongs to. The second number denotes the column that this entry belongs to. Thus 2,3 is the entry in the second row and third column. It does not in any way relate to a decimal fraction $2\,^{3}/_{10}$.

We are also told that: "Entries in the j-th column have been in DIAPAUSE for j days". This means that the particular column of an entry (first, second, third, j^{th}, etc.) of an entry tells us the number of days those eggs have been in DIAPAUSE. A number - say 666 in column 9, means that six-hundred-and-sixty-six eggs have been in diapause for nine days.

Further, "(1,j) = quantity of eggs laid in wet conditions, in DIAPAUSE for j days". This means that the entire first row denotes eggs laid in wet conditions. (1, j) means row 1, and j means the jth day. So, the column number tells how many days the eggs have been there, the row number tells whether they were laid in wet or dry conditions.

The DIAPAUSEA variable would typically look like this:

```
0     0     100
0     500   0
```

In this example, there are 500 eggs that were laid in dry conditions, and have been in DIAPAUSEA for 2 days. There are 100 eggs that were laid in wet conditions, and which have been in DIAPAUSEA for 3 days. There are no eggs at all that have been in DIAPAUSEA for only 1 day, and none that have been in DIAPAUSEA for longer than 3 days.

In the ICS_ file, the variables are not written in several rows as shown above, but they do adopt the structure as shown. The difference is that a new row is signified by a semi-colon" ;". Hence the ICS_ file would represent the above conditions by the statement

```
DIAPAUSEA = [0 0 100;0 500 0];
```

NOTE: Semi-colons demarcate new lines. Note there is a semi-colon after the first three numbers, as well as at the end of the statement.

The initial condition does not necessarily have to contain a history of 3 days (or whatever number of days). It could simply be numbers of eggs for the starting day, in which case it would have only one column (n = 1). HOWEVER, it is required to have two rows, no matter whether a history is included or not.

```
DIAPAUSEA = [0;0];
```

denotes that on the start day there were no diapause eggs (column one is zeros), and before the starting day there is no knowledge of the diapause eggs number (since there is no second, third, etc. column).

A typical ICS_ file might look something like this:

```
ADULTMALESa        =[00];
LAY1a              =[00;0];
LAY2a              =[00;0];
LAY3a              =[00;0];
LAY4a              =[00;0];

DIAPAUSEa          =[100;0];
DIAPAUSE1a         =[100];
DIAPAUSE2a         =[100];
NONDIAPAUSEa       =[100;0;0];
TURGID1a           =[100;0;0];
TURGID2a           =[100;0;0];
QUIESCENT1a        =[100];
QUIESCENT2a        =[100];

INSTARSa           =[00;0];
MATURINGADULTa     =[00;0];
```

NOTE: The spaces on either side of the "=" symbol are optional. They serve to align the numbers for appearance sake only. Blank lines between sections are optional.

NOTE: Double zeros "00" are interpreted as numeric zero "0", the two notations are interchangeable.

NOTE: The interpreter is case-sensitive. "INSTARSa" is NOT the same variable as "instarsa", or "Instarsa". Failure to interpret the data will result if cases are switched.

NOTE: You can have many initial conditions files stored on disk. Only the one with the DOS file name ICS_.M, where "_" corresponds to the station (a, b, c, etc.) will be interpreted. Name your files in any convenient way, and simply change the name of the file you want to be interpreted as the initial conditions for a particular run. In fact, you don't even need to rename files, simply call the appropriate file from the Matlab command window!

WHEN ARE INITIAL CONDITIONS USED?

The initial conditions are read during the execution of **load data** , or if the **reset ICs** option is called from the **tools** section of the Control Panel. They are also read if the instruction **icsa** ↵ or **icsb** ↵ is issiued at the Matlab command prompt.They are not read again or interpreted again during any of the Simulink procedures (procedures launched from the model block diagram window).This has important implications!

Suppose you load weather data, and start a simulation run. Then you might examine the results and decide to re-run the simulation, after possibly modifying some of the lifecycle parameters. If you were to commence a second simulation run, by choosing **Simulation / Start** from the Simulation Block Diagram Window, an interesting situation would arise.

The initial conditions for the second run would be those values that were 'left' from the first simulation run. In other words, simulation-run-two's initial conditions are the ending condition of simulation-run-one. This is fine if your two consecutive simulations are based on weather data for consecutive periods, but results in meaningless results if you wish to re-simulate the same period. To re-simulate the same period, or any non-consecutive period, it is necessary to load the correct initial conditions.

This can of course be done by starting afresh, and selecting the **load data** pushbutton. This, however, performs a number of calculations that do not need to be redone, and is wasteful of time. The initial conditions contained in the disk file can be manually reloaded prior to any simulation run by typing **icsa** ↵ at the Matlab Command Prompt ">>", to reload for station A. Type **icsb** ↵ to reload initial conditions for station B, **icsc** ↵ to reload initial conditions for station C, and so on. A quick way to reload ALL the initial conditions is to choose the **tools / reset ICs** option sequence from the Control Panel menu bar. Note that this resets the initial conditions for all stations.

NOTE: You need to reload initial conditions for **all** the stations, A, B, C etc. Only under exceptional circumstances might you need to reload initial conditions for some stations and not for others.

See **Appendix G** for a detailed explanation of each population variable's structure.

Appendix E Section 5 Customising the model parameters

SIMULATION MODEL STRUCTURAL OVERVIEW

The simulation model can be viewed as a suite of hierarchically nested structures. A top-down approach has been used , meaning that large-scale concepts are considered first, progressing to finer detail as it becomes necessary.

Hierarchically, the model may be represented:

N-nodemodel of sites

⇩

Lifecycle model for n^{th} node

⇩

Lifecycle stages

⇩

Application of the rulebase

This structure is represented diagramatically in figure E5.1, overleaf. This diagrammatic representation is identical to that adopted in the Simulink representation of the model. That is, this visualisation as a suite of nested structures has been carried right through to the software implementation of the model

In order for therulebase to make appropriate decisions on the branch-points of the lifecycle model, it needs information concerning those parameters which govern transitions between different stages in the lifecycle.

EXAMPLE:

A certain proportion of fifth-instar hoppers might not progress to becoming maturing adults, due to predators, shortage of resources etc. The fraction of the individuals that do survive needs to be known by the rulebase, so it can transfer the appropriate number of individuals from the intar stage to the adult stage. This information is supplied to the rulebase in the form of a lifecycle parameter, called in this case, "survivalrate". Survivalrate is exclusive to the INSTARS lifecycle stage. Other lifecycle stages certainly have survival rates, BUT they do not affect this survival rate in any way.

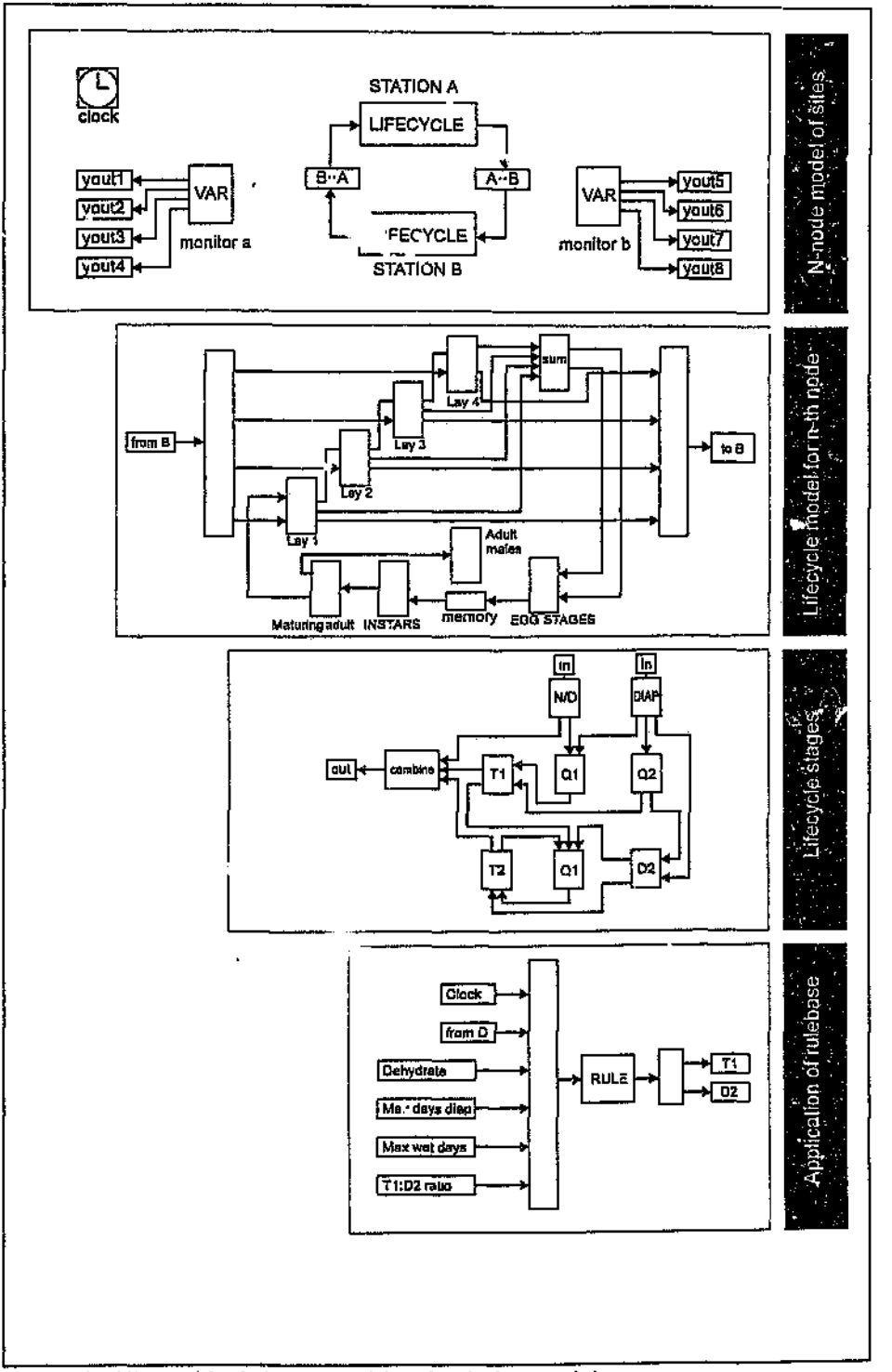


Fig E5.1 Hierarchical structure of the simulation model

CHANGING THE NUMERICAL VALUES OF THE LIFECYCLE PARAMETERS

NOTE: You need to have started Simulink to alter the parameters. To do so, choose the "Simulate" pushbutton from the Control Panel window, or type eggcyc09 ↵, at the Command Prompt. The N-site block diagram will be displayed. You do not need to actually start the simulation run!

Decide which for which site you would like to alter the lifecycle parameters. You can change the parameters for each site independantly of any other site. Double left-click the mouse pointer "↖" on the block that represents the station of your choice. Such blocks will be labelled "LIFECYCLE" within the block, and have an associated identifier "STATION A" or "STATION B".

NOTE: The blocks denoted "MONITOR" and "yout"1···yout8 contain routines necessary to extract population data from the lifecycles. You should not edit these blocks, nor should you have need to access them at this stage!

Once you have clicked on the lifecycle block, a Lifecycle Window will open in the foreground. See figure E5.1 for typical examples.

Choose for which particular lifecycle stage you wish to alter the parameters. Such stages will typically be labelled: "Lay 1", "Maturing adult", "EGG STAGES". Select the chosen block by double left-clickingthe mouse pointer on that block.

NOTE: EGGSTAGES is different from the other blocks in this window. EGGSTAGES contains one hierarchical level more than any other block.

The Lifecycle Stage window will open, corresponding to the particular lifecycle stage block you have clicked on.

If you chose any stage other than EGGSTAGES, double left-clicking the mouse pointer "↖" on that lifecycle-stage block will display the lifecycle parameters associated with that stage. The available lifecycle parameters, along with their recommended values, are listed in Appendix G. To alter any parameter, simply double left-click the mouse pointer "↖" on the block containing the parameter. A dialogue box willopen, allowing you to edit the numeric value of the parameter. See figure E5.2 for a typical example.

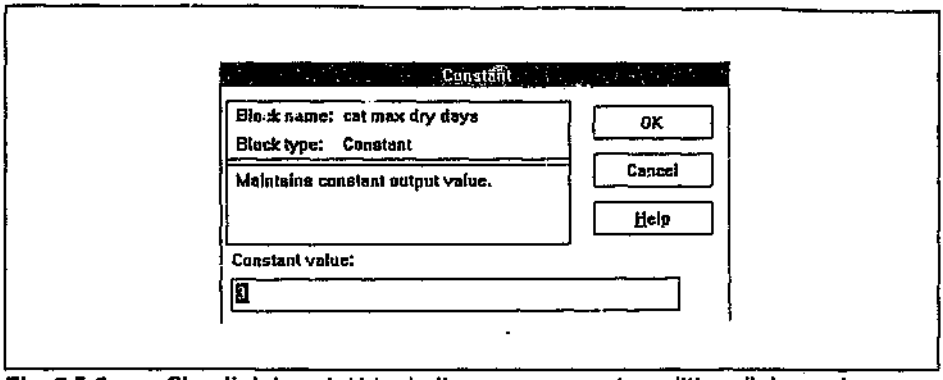


Fig. E 5.2 Simulink / model block diagram parameter editing dialogue box

⊞: When the current value of the parameter is first displayed, it will be highlighted. If you press the **backspace** key, the number will be erased in anticipation of you entering a new one. However, if the parameter involves a two- or three digit number, for instance 100, that you wish to change to 105, it is more convenient to press **right cursor**, and then **backspace**. This will only erase the last digit, and you only need to enter the changed digit.
It is more convenient to press ↵, rather than clicking on the **OK** pushbutton, once the value is adjusted.

If you have chosen EGGSTAGES as the lifecycle stage you wish to edit, there is an additional step involved. Double left-clicking the EGGSTAGES block will display an intermediate block diagram which shows all the egg stages. To alter a lifecycle parameter associated with any one of the egg stages, simply double left-click on the block corresponding to the egg stage of your choice (for example, turgid1, quiescent1, diapause, etc.)

NOTE: The blocks denoted "In_1"," In_2" and "Out_1" are used to control the flow of information between hierarchical levels **and should not be edited!**

NOTE: You may change lifecycle parameters while the simulation is running. If you change a value, the new value is used for all simulation days after the day on which you changed the parameter. This can lead to confusion, and results that are difficult to repeat.

NOTE: If you change a parameter, you will need to update the disk file in which the lifecycle model is stored. Otherwise your changes will remain in effect only during the current session. To save your changes, select the File option from the Simulink Block Diagram Window (the window that displays the N-site simulation block diagram). From the drop-down menu (New / Open / Close / Save / Save As / Print / Printer Setup / Exit), choose the Save option.

If you select the Save As option, you will be required to choose a new file identifier for the block diagram data, and you will also have to edit the program code which references that data (references to this data are to be found in the script file SIMOPTS.M). If you are not confidant that you can edit the code, DO NOT use the Save As option.

If there are certain sets of different lifecycle parameters that you will use frequently, you can add extra menu items or pushbuttons to the SIMULATION RUN OPTIONS dialogue box. These can be used to quickly invoke one of several simulations, which differ only in their lifecycle parameters. In this way, you would not have to re-edit lifecycle parameters each time you wanted to use a different (but often used) set.
For example, you could have two "Simulate" pushbuttons:
① simulation where the adult female laying interval is 7 days.
② simulation where the adult female laying interval is 10 days.

NOTE: When you save a modified block diagram, all the current parameters and block properties are updated. These include properties such as the names of blocks, the interconnections between blocks, block positions on screen, and so on. If you have changed any of these parameters, the changes will replace the original properties when you re-open the block diagram during a future session.

You can now return to the two-site (or n-site, if you have an upgraded version) representation of the model, and commence a simulation run, by choosing simulation and thence run, from the Simulink/ eggcyc09 window menu bar.

Your updated and modified parameters will take effect during all subsequent simulation runs.

NOTE: It is possible to enter a lifecycle stage block and alter parameters while a simulation is actually running (day counter incrementing indicates this). However, such changes will have unpredictable, and unrepeatable results, since it is not posssible to determine on exactly which day of the imulation the change came into effect.
For this reason, all changes to lifecycle and migration parameters should be made while the simulation is not running.

The technical description of the rulebase (appendix D) gives a concise and functional description of how each lifecycle parameter is used within the model. Refer to this section to learn how changes will be interpreted by the rulebase.

CHANGING S USED IN WEATHER DATA PRE-PROCESSING

When weather data is loaded initially, certain pre-processing is performed, to create intermediate variables which are used by the simulation in place of the actual weather data fields. Air and soil teperatures are used to compute degree-day contributions, rainfall is used to create a wetindex, and wind speeds and headings are used to create a windindex.

The parameters used in these calculations are not available from the block diagram of the model, and can only be adjusted by editing the Matlab script files with a ditor.

THE DEGREE-DAY CONTRIBUTION CALCULATION

This calculation is performed in a script file called PRECALC_.M where "_" is either "A" or "B" or "C" etc.and denotes the site for which the calculation is being performed. Thus the rules for degree-day computation can be different for site A and for side B. For a generic change to be effected, all relevant PRECALC files must be edited to reflect the same rules. PRECALC_.M is automatically called during the data loading procedure.

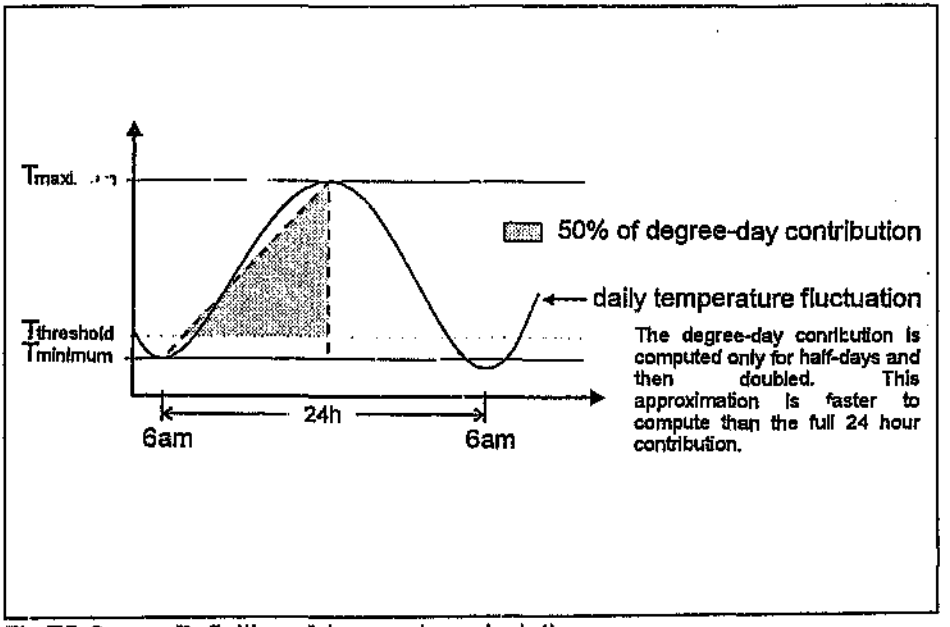


Fig E5 .3: Definition of degree-day calculation

The only user-adjustable entry is the lower threshold for development. This value may be different for soil temperatures - which determine egg stage development, and air temperatures - which determine hopper stage development. The value specified denotes the temperature below which all development will cease. Temperatures below this value offer no degree-day contribution.

NOTE: This value overrides any value specified in the data loading dialogue! If the value in the data loading dialogue is to be used, then there should be no reference to lowerthreshold in any of the PRECALC_ scriptfiles.

To alter the override value, locate the line:

lowerthreshold = 20;

and alter the value. (units are °C)

NOTE: There can be two instances of **lowerthreshold** assignment statements. The first is for the degree-day air contribution, the second for the degree-day ground contribution. The same variable name is used, but its value is adjusted prior to each degree-day calculation.

NOTE: Remember to save the changes you make to the file. Since the precalculation procedure is only called during the weather data loading phase of the simulation, changes made here will not take effect until the weather data is reloaded from disk.

THE WET-INDEX CALCULATION

The wet index is used extensively by the decision rule engine - the reader is refered to the rulebase summary to find out how components of the wetindex are interpreted by the rules.

A wet index is created for each site, and is denoted WETINDEXa, WETINDEXb, etc. WTINDEXa is generated in PRECALCA.M, WETINDEXb is created in PRECALCB.M etc.

NOTE: WETINDEXa, WETINDEXb are Matlab variables (matrices), stored in memory. PRECALCA.M, PRECALCB.M are script files - stored on disk.

To adjust the rules and parameters used to compute WETINDEX_, you need to open and edit the correspondinf Matlab scriptfile with a text editor.

Wetindex is a matrix with the following structure:

rows X columns = n X 3, where n is the numver of days for which weather is available (same number of rows as the weather data file).

Column1 of WETINDEX_ is a binary (0 or 1) flag indicating if that day is wet (1) or dry (0). A day is wet if the rainfall on that day is non-zero, however a day is not necessarily dry even if the rainfall on that day is zero. A sufficiently large fall of rain on a particular day can make several subsequent days 'wet' even if no rain falls on those subsequent days.

The following rules apply:

Rainfall on current day	Number of subsequent days -Including today- regarded wet
0	0
$0 < R^{mm} \leq 6$	1
$6 < R^{mm} \leq 10$	2
$10 < R^{mm} \leq 15$	3
$15 < R^{mm} \leq 25$	4
$25 < R^{mm}$	5

Column 2 is computed from entries in column 1. Column 2 denotes the number of censecutive wet days, or consecutivbe dry days that have been experienced up until and including the current day. Column 2 contains an integer number, positive values (>0) are used to denote n consecutive wet days. Negative numbers (<0)

are used to denote n consecutive dry days. For example, if it has been wet for 5 days already on a particular day, column 2 of WETINDEX for that particular day will be +5. However, if it has been dry fo 5 days on a particular day, the column2 entry of WETINDEX for that particular day will be -5.

Column 3 of WETINDEX_ represents the number of consecutive rainy days up to and including that particular day. If no rain falls on a particular day, then that day's column 3 entry will be zero. A day is considered 'rainy' if rain fell on that day.

EDITING THE CLASSIFICATION BOUNDARIES

There is no anticipated need for the user to change anything except the delimiting values used to determine how many wet days follow a rainy day (column 1). These values are changed within the source code for PRECALCA.M or PRECALB.M etc. The appropriate code segment is shown below (from PRECALCA.M), the boldface numerals are those that the user may wish to change.

Note: These must be changed in pairs, except for the zero value, and the maximum value (25mm).

```
for day = 1:length(RAINFALLa),
  if ((RAINFALLa(day) > 0) & (RAINFALLa(day) <= 6))
   WETINDEXa(day,1) = [1];
  end;
  if ((RAINFALLa(day) > 6) & (RAINFALLa(day) <= 10))
   WETINDEXa(day:day+1,1) = [1;1];
  end;
  if ((RAINFALLa(day) > 10) & (RAINFALLa(day) <= 15))
   WETINDEXa(day:day+2,1) = [1;1;1];
  end;
  if ((RAINFALLa(day) > 15) & (RAINFALLa(day) <= 25))
   WETINDEXa(day:day+3,1) = [1;1;1;1];
  end
  if (RAINFALLa(day) > 25)
   WETINDEXa(day:day+4,1) = [1;1;1;1;1];
  end;
end;
```

NOTE: Remember to change PRECALCA.M and PRECALCB.M and all other PRECALC_.M files that are available. i.e., one for each site in use.

THE WIND- INDEX CALCULATION

The variable WINDINDEX is common to all sites. The WINDINDEX is created by a script file called WINDCALC.M, and there is no distinction between sites. Internally, WINDINDEX reserves separate columns for different sites. WINDCALC. M is automatically called during the data loading procedure.

WINDINDEX is the key to the migration model. WINDINDEX contains data that represents the probability that insects will be carried from one station to any other station, and a brief description of these probability functions will be given here.

The insects are assumed to be carried at the prevailing wind speed, v, for a time period, t, controlled by the prevailing air temperature.

The distance travelled is then s, where $s = v \cdot t$

MIGRATION DISTANCE PROBABILITY

The model operates only at a few points in space, not an infinitley resolvable continuum. Therefore it is not useful to determine an absolute destination for the migrating insects, since that may well lie at some point other than where a population simulation is located. However, it is possible to consider only the descrete locations, and estimate a probability that locusts will be deposited there, keeping in mind that a locust swarm is a nebulous entity, and that the wind measurements are only applicable at one instant during the day. The variation in wind speed and direction, and the fuzzy boundaries of swarms make the destination of insects a region, rather than a point. The probability approach allows this cloudiness to be modelled in an analogous manner to the use of wave probability functions in particle physics.

If the migration distance, s, is known, and the interstation distance,s', is also known, then we can calcultate the probability that these two values coincide. If s = s', then the probability, P, is 1, i.e., we assume that there is a 100% chance that the locust will be translated between the two sites (direction assumed to be correct for the time being).

A Gaussian probabilty distribution is used, meaning that P is never equal to zero, but approaches some assymptotic value as the difference between s and s' approaches infinity.

The actual calculation used is:

$$P = e^{-b(s-s')^2}$$

where $b = \dfrac{\ln_e(100)}{(s')}$

As s become more and more remote from s', so P tends towards zero. The exact shape of the curve can be altered by varying the value of the constant b.

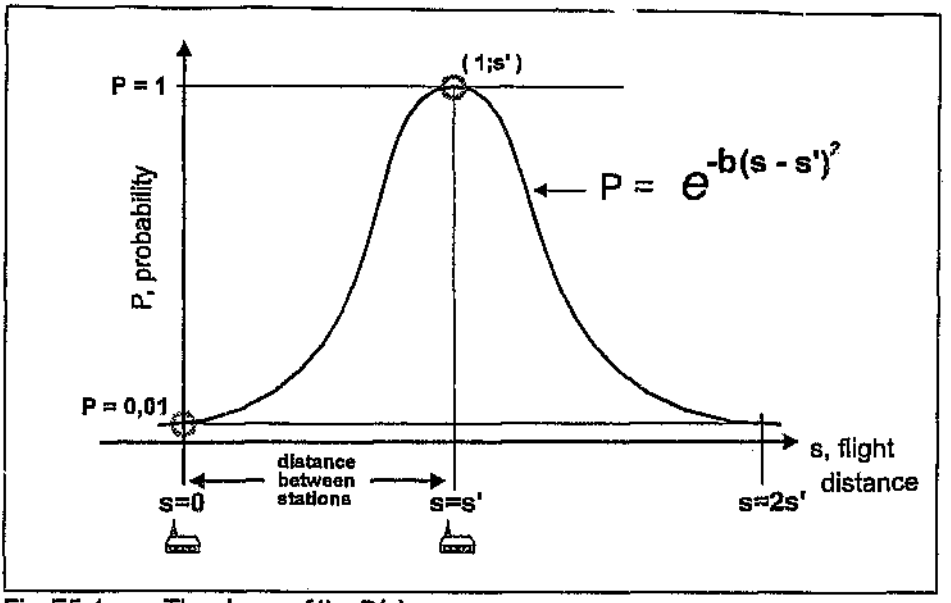


Fig E5.4: The shape of the P(s) curve

These probabilities are stored in columns 1 & 4 of WINDINDEX, for sites A and B respectively.

FLIGHT TIME ESTIMATION

The flight time is estimated from the change in air temperature. This sinusoidal fluctuation is approximated by a triangular wave function, forced to fit through the measured maximum and minimum daily air temperatures.

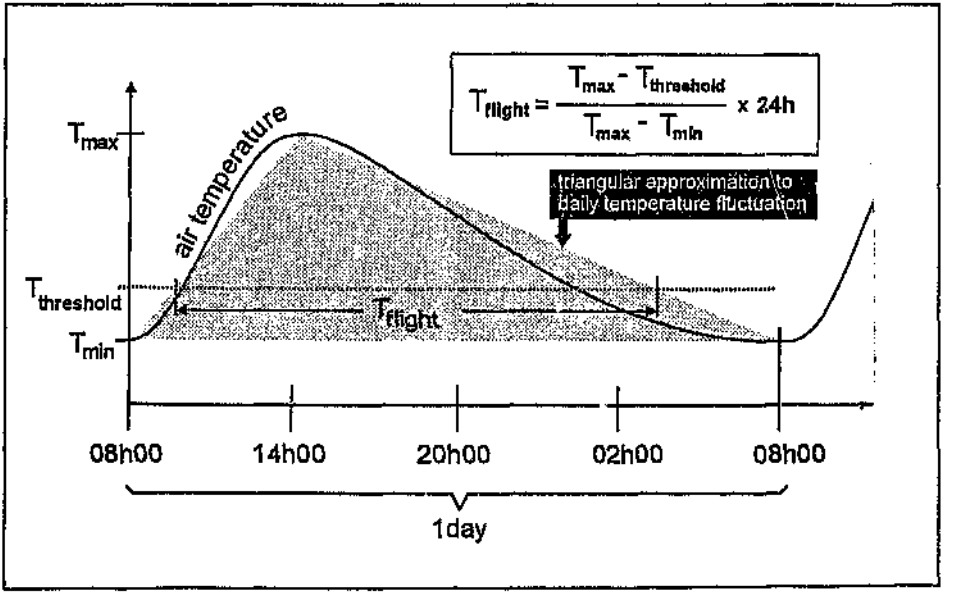


Fig E5.5. Flight time estimation curve

The period during which the air temperature remains above the threshold temperature for flight is considered to be the flight time. This is first computed as a fraction of the day - since this is more conducive to the geometry of the diagram, and then translated into a time in hours. The threshold temperature is specified by the constant assignment

Tthresh = 18

in the file WINDCALC.M. Units are °C. Higher threshold values will result in shorter migration periods..

If the flight threshold temperature is very low, or the air temperature is very high, the air temperature may remain above the threshold for the entire 24hour period. In this case some other factor is necessary to cause cessation of flight. In reality, this is often some function of the insect fat reserves. No measure of this available to the model, and hence an alternative method has to be sought . truncate the flight time. At present, flight time is limited by a constant named **maxhours**, indicating the maximum posible hours of flight, when temperature is not a limiting factor. maxhours is defined within WINDCALC.M, by an assignment statement of the form:

maxhours = 20;

The user may wish to alter this to any other appropriate value, although numbers larger than 24 will result in meaningless, or contradictory calculations.

NOTE: Remember to save the changes you make to the file. Since the wind calculation procedure is only called during the weather data loading phase of the simulation, changes made here will not take effect until the weather data is reloaded from disk.

WIND DIRECTION PROBABILITY

In the same way that a probability function is used to determine the likleyhood of a certain migration distance matching the interstation distance, a probability function is used to determine how closely the wind heading matches the relative bearings of the sites under consideration.

We have already described how the probability relating flight distance to interstation distance is computed. This description assumed that the migration was occuring in the correct direction, i.e., directly along the line joining the two stations.

If the wind direction is along that line, then the assumption is valid. If the wind direction is something other than along that line, then there is a less than 100% probability that locust will move from one station to the other. The probability maps the actual relative bearings of the stations being considered to the daily wind heading. A Gaussian probability distribution is used, meaning that P is never equal to zero, but approaches some assymptotic value as the difference between the two bearings approaches 180°.

The computation used is:

$$P_{a\text{-}b} = e^{-a(\delta-180^\circ)^2}$$

where $a = \dfrac{\ln_e(100)}{(180^\circ)^2}$

For any pair of weather stations, it is convenient to define two pairs of probability functions:

P_A(A to B) probability of getting insects from A to B, using wind data at A
P_A(A from B) probability of getting insects from B to A, using wind data at A

P_B(B to A) probability of getting insects from B to A, using wind data at B
P_B(B from A) probability of getting insects from B to A, using wind data at B

These probabilities are stored in WINDINDEX columns 2,3,5 & 6 respectively.

Changing the shape of the curve is effected by altering the value of the constant a, wihin the file WINDCALC.M.

COMBINING PROBABILITIES

The three probabilities for each site are combined. We have computed for WINDINDEX:

$P_A(s)$	column 1	the probability that locusts will be carried from one site to the other, based on the wind speed at A, and the interstation distance.
$P_A(A \rightarrow B)$	column 2	the probability that locust will be transported from A to B, based on the wind heading at A
$P_A(A \leftarrow B)$	column 3	the probability that locust will be transported from B to A, based on the wind heading at A
$P_B(s)$	column 4	the probability that locusts will be carried from one site to the other, based on the wind speed at B, and the interstation distance.
$P_B(B \rightarrow A)$	column 5	the probability that locust will be transported from B to A, based on the wind heading at B
$P_B(B \leftarrow A)$	column 6	the probability that locust will be transported from A to B, based on the wind heading at B

These are combined by simple multiplication of the probabilities:

$P(A \rightarrow B))$	column 7	$k_1 \cdot P_A(s) \cdot P_A(A \rightarrow B) \cdot P_B(B \leftarrow A)$ (columns 1,2,6)
$P(B \rightarrow A))$	column 8	$k_2 \cdot P_B(s) \cdot P_B(B \rightarrow A) \cdot P_A(A \leftarrow B)$ (columns 4,5,3)

k_1 and k_2 are simple scaling constants, determined empirically; they adjust the magnitude of the migratory fraction of the population to bring it as closely in line with the observed events as possible. Values of »5 have been found to work well.

EMMIGRATING POPULATION ESTIMATES

So far, what has been computed will aid in estimating what fraction of individuals that leave a station will arrive at other stations. In order to etimate the magnitude of the emmigrating swarm, the wind speed and the temperature at that site are monitored.

Emmigration is only allowed if the maximum air temperature is above the threshold temperature for flight. If this is not the case, emmigration is inhibited. Adjustmwent of the threshold temperature for flight is described in the fight time estimation section above.

In order to estimate exactly what proportion of a population will emmigrate, based on wind speed data, the maximum occuring wind speed is considered. If an assumption were to be made that a certain wind speed would result in a migration of a certain magnitude, there could arise problems when large wind speeds occured. It is conceivable that high wind speeds would theoretically remove a larger number of insects than existed in the population. To alleviate this potential problem, and still retain a linear contstand of proportionality between wind speed and size of migration, the largest occuring wind speed is assumed to remove the entire population, or if desired, some large fraction of the entire population.

A variable, depopwind, is generated equal to the largest occuring wind speed. All daily wind speeds are compared to that maximum value, and a proportional fraction of insects is considered to be the emmigrating fraction.

If it is desired that the maximum occuring windspeed only removes a fraction of the population, then that fraction may be included in the depopwind assignment statement in WINDCALC.M. For example, the statement:

depopwind = max([max(WINDa(:,1)) max(WINDb(:,1))]);

causes the maximum occuring wind speed to remove 100% of the population. However, if it is required that this wind speed only remove 75% of the population, then the assignment should be changed to read:

depopwind = (100/75) * max([max(WINDa(:,1)) max(WINDb(:,1))]);

NOTE: The boldface characters indicate the necessary changes. The value of 75 can be changed to any percentage of total population that the user desires.

On each day, the daily wind speed is compared to the maximum wind speed, and that ratio (corrected or uncorrected, as desired) is used to estimate the proportion of the population emmigrating. The ratios of actual wind to maximum wind speed are stored in columns 9 and 10 of WINCALC, for sations A and B, respectively.

$P_{emmigrate/A}$	column 9	daily wind speed / maximum ocurring wind speed at station A
$P_{emmigrate/B}$	column 10	daily wind speed / maximum ocurring wind speed at station B

Appendix E Section 6 Adding your own or updated weather data

BASIC PROCEDURE

If you are simply adding updated weather data for a town or station that the simulation model already recognises, all that is required is to replace the old weather data file with the newer file, using the same file name. Data files are named according to the following convention:

The first two letters of the file identifier represent the town code. dg = Douglas, gr = Graaff-Reinett, pa = Pofadder, mb = Middelburg, and so on. The next four characters denote the data fields present in the file: a = air temperatures, s = soil temperatures, r = rainfall, w = wind speed and direction. The file extension is always ".m" for ASCII text files, and "wk1" for Lotus™ 1-2-3 spreadsheet files. Hence the file **dgasrw.m** contains air temperatures, soil temperatures, rainfall and wind data for the town of Douglas, in ASCII data format. **dgasrw.wk1** contains the same information, but in Lotus 1-2-3 WK1 format.

Adding a new, previously unrecognised station requires - in addition to creating the data file - editing the simulation model's script files. A new data file has to be created - according to the rules defining the format of the files. Further, the co-ordinate data for the station has to be made available to the model (latitude and longitude values). Finally, the name of the new station has to be entered into the drop-down menu, so the new station can be chosen as a site during a simulation. Editing if the necessary files is described later in this chapter.

WEATHER DATA REQUIRED

The simulation model requires the following climatic data:

① maximum daily air temperature
② minimum daily air temperature
③ maximum daily soil temperature
④ minimum daily soil temperature
⑤ daily rainfall
⑥ daily wind speed at time of locust flight
⑦ daily wind bearing at time of locust flight

TWO TYPES OF DATA FILE ARE POSSIBLE

This data can be read from a DOS ASCII text file, using the ASCII #10 #13 CR/LF sequence as an end-of-line marker. Most spreadsheet print-to-file options and text editors can produce output of this format.

The Simulation Model can also read the weather data directly from a Lotus™ 1-2-3 spreadsheet file. Lotus 1-2-3 saves its spreadsheets in data files with the extension ".wk1". Many other spreadsheets and database programs (such as Quattro-Pro for Windows, Q&A for Windows) allow you to save data in this format. You can create and edit data in your favourite spreadsheet, or import data from

any other source that your spreadsheet supports, and simply save the spreadsheet as you would normally do. The simulation model can then access this file, without the need to convert it or process the data again.

NOTE: You do not have the option of selecting one data file type or the other. The data file format is at present inherent in the simulation model source code. Make sure that the version you have supports the type of data file you wish to use. Adjustment of the file type can be performed by editing the script file LOADDATA.M.

Shown below is a short extract from LOADDATA.M, illustrating the segment that reads data for the town of Jansenville. As shown, the code segment reads only WK1 format data.

```
%       JVASRW;
        DATA = wk1read('JVASRW',-3,0);
        town = 'Jansenville';    %%%
```

To alter the above code to make it compatible with ASCII text data files, remove the "%" symbol preceding the ASCII data file name "JVASRW". Next, insert a "%" symbol in front of the "DATA" statement, thus:

```
        JVASRW;
%       DATA = wk1read('JVASRW',-3,0);
        town = 'Jansenville';    %%%
```

The above code will now interpret ASCII data files only.

You can mix different file types and their calls within LOADDATA. You only need to change those data file calls for which you have only ASCII data files. For example, if you only had Jansenville's data in ASCII format, you could perform the change shown above, and then save the file. All weather data other than Jansenville would be required in WK1 format. If all you files are available in only ASCII format, then you need to perform changes outlined above for every station listed within LOADDATA.M

You may also wish to incorporate actual recorded locust count data in the data file. In this way it will be possible to plot the recorded numbers on the same axes as the simulated results, for a convenient comparison.

ASCII TEXT FILE FORMAT SPECIFICATION

NOTE: The following rules apply to the structure of the weather data file. Deviations from this format will result in failure of the simulation model to read the data at all, or may result in data being misinterpreted by the model. In this case there may be no warning that the data is being processed incorrectly, and the simulation result may be meaningless.

☆ MSDOS or compatible ASCII text file (7-bit) with #10 #13 CR/LF end-of-line delimitation.

- ☆ Data segment width is 40 columns. Although more columns can be added, they will be ignored unless additional code is written to take advantage of these columns.
- ☆ Number of lines is unlimited, dependant only on the amount of RAM available during loading of the data into the Matlab workspace.
- ☆ Each line (row) represents data for one day. The data on the su' sequent line corresponds to data applicable to the following day.
- ☆ The data file may commence with a user-specified preamble. This may contain any information that you would like to incorporate to help identify the data, or explain its use. Such a preamble might typically specify the starting date of the data, the ending date, the site for which it was recorded, and possibly a key explaining the content of each field.
- ☆ Any such non-data preamble must be "commented" by means of a percent symbol "%" in column 1 of that line. Any number of commented lines may precede the data segment, but once the data segment has begun, no comments may come within the data area.
- ☆ The data segment is start-delimited by the character string: **DATA=[...**
- ☆ The data segment is terminated by the character string: **]**
- ☆ The individual fields of the data segment are delimited by one or more consecutive SPACE characters (ASCII #32).
- ☆ The end of the data file is marked with the standard EOF marker (^Z).
- ☆ Each line may contain the semi-colon character ";" in the last column. The CR/LF sequence is still required to indicate the end-of-line. Placing a ";" in the last column of each line will suppress the display of weather data in the Matlab Command Window when that data is being loaded into the workspace. Since it must be the last character in a line, no characters may be placed to the right of the ";".
- ☆ All lines must contain the same number of fields. No field may be left empty. Any field within the data block must contain numeric values. If no data is available for any particular field, that field must be filled with a dummy number, possibly 999, or 0, or -1. Select a dummy number that is easily identified (such as 999) so it will not be confused with real data.
- ☆ The left-most field in each row must contain a three digit day reference, or be filled with a series of dummy numerals. No actual weather data may occupy the first field.

ASCII DATA FILE EXAMPLE

An extract of the first 10 lines of a weather data file is shown below.

```
% Douglas: Aug-28,1995..Dec-31,1997
%ID Amx Amn Smx Smn Rmm a/s h/s 10a <8a 12p <2p
DATA=[...
240 275  55 250 125   0   0   3   0   0   0   0
241 270  55 245 125   0   0   2   0   0  50 310
242 275  60 235 130   0   0   4   0   0  50 30
243 270  60 250 120   0   0   3   0   0   0   0
244 275 100 265 145   0   0   3  50  90   0   0
245 290 115 260 150   0   0   3  50  90 100 360
246 270  95 225 155  78   0   2   0   0 100 230]
```

⇦ Comment line, note % in first column.
⇦ Additional comment line, identifies fields.
⇦ Start of data segment identifier
⇦ First line of data segment. Fields are delimited by space characters. Each line represents data for a single day

⇦ Last line of data segment. Note that end of data segment is marked by a "]" character. The EOF end-of-file marker is non-printable, and is not normally displayed.

ASCII DATA FILE FIELD SPECIFICATION

The data file is required to be not less than 40 columns wide. Additional columns may exist, but they will not be processed by the interpreter.

NOTE: A zero may precede any numeric value that is recorded to only one or two significant digits, in order to preserve the visual symmetry of the file.
For example, 5mm of rainfall may be entered as " 5", i.e., "SPACE SPACE 5", or it may be entered as "005", i.e., "ZERO ZERO 5". Either of these is acceptable.

Column(s)	Data description
1, 2, 3	Day Identifier. Three digit number denoting the day of the year. January 1st is day number 001, December 31st is day number 365 (non-leap years) and day number 366 (leap years only).
4	space character.
5, 6, 7	Maximum air temperature. Three digit number indicating the maximum air temperature measured in °C for that day. A decimal point is assumed to lie between column 6 and column 7, hence an entry "275" is interpreted as 27.5°C. A "-" in column 5 indicates a negative temperature, below freezing.
8	space character.
9, 10, 11	Minimum air temperature. Three digit number indicating the minimum air temperature measured in °C, for that day. A decimal point is assumed to lie between column 10 and column 11, hence an entry "95" is interpreted as 9.5°C. This value can also be entered as "095". A "-" in column 9 indicates a negative temperature, below freezing.
12	space character.
13, 14, 15	Maximum soil temperature. Three digit number indicating the maximum soil temperature measured in °C, for that day. A decimal point is assumed to lie between column 14 and column 15, hence an entry "275" is interpreted as 27.5°C. A "-" in column 13 indicates a negative temperature, below freezing.
16	space character.

17,18,19	Minimum soil temperature. Three digit number indicating the minimum soil temperature measured in °C, for that day. A decimal point is assumed to lie between column 18 and column 19, hence an entry "75" is interpreted as 7.5°C. A "-" in column 17 indicates a negative temperature, below freezing.
20	space character.
21,22,23	Rainfall. Three digit number indicating the rainfall measured in millimetres, for that day. No decimal point is assumed. Negative numbers are meaningless, and will be interpreted as zero rainfall.
24	space character.
25,26,26	Recorded numbers of adult swarms. Three digit number indicating the number of adult swarms observed on that day. This data is not used by the simulation. Plotting these data might provide useful comparisons. The format and structure of this field is entirely user-specified, **HOWEVER, the field may not be left blank.** If no data is available for this field, the columns must be filled with zeros, or any other numeric values.
27	space character.
28,29,30	Recorded numbers of hopper bands. Three digit number indicating the number of hopper bands observed on that day. This data is not used by the simulation. Plotting these data might provide useful comparisons. The format and structure of this field is entirely user-specified, **HOWEVER, the field may not be left blank.** If no data is available for this field, the columns must be filled with zeros, or any other numeric values
31	space character.
32,33,34	Wind magnitude at 08h00. Three digit number indicating the wind speed magnitude measured in knots, at 08h00 on that day. (one knot = one nautical mile per hour) A decimal point is assumed to lie between column 33 and column 34, hence an entry "75" is interpreted as 7.5knots. Negative speeds are meaningless but a "-" in column 32 will be interpreted to mean that the wind direction is opposite to that specified in the "direction" field.
35	Space character.
36,37,38	Wind heading (direction) at 08h00. Three digit number indicating the wind heading measured in decimal degrees at 08h00, for that day. No decimal point is assumed. Angles are measured relative to true north, in a counter-clockwise direction. A "-" in column 36 will be interpreted as a negative angle, which corresponds to a positive angle of the same magnitude but measured in a clockwise direction. Negative angles should be avoided, since values larger than 99 cannot be accommodated. Valid angles fall in the range 0···360° inclusive. Note that the wind heading is the **direction from which the wind blows**, not the direction to which it blows. Hence a wind heading of 0° implies the wind is northerly, or blowing towards the south.
39	Space character.
40	Optional semi-colon ";".
41···	Ignored.

NOTE: The following rules apply to the structure of the weather data file. Deviations from this format with result in failure of the simulation model to read the data at all, or may result in data being misinterpreted by the model. In this case there may be no warning that the data is being processed incorrectly, and the simulation result may be meaningless.

☆ Lotus 1-2-3 .wk1 format spreadsheet file, or compatible

☆ Data segment width is 10 fields (cells). Although more fields (cells) can be added to a row, they will be ignored unless additional code is written to take advantage of these fields.

☆ Number of rows is unlimited, dependant only on the amount of RAM available during loading of the data into the Matlab workspace. Automatic row counting is supported.

☆ Each row represents data for one day. The data on the subsequent row corresponds to data applicable to the following day.

☆ The data file MUST commence with a 3-row user-specified preamble. This may contain any information that you would like to incorporate to help identify the data, or explain its use. Such a preamble might typically specify the starting date of the data, the ending date, the site for which it was recorded, and possibly a key explaining the content of each field.
These three rows may be left blank, or contain any character strings you like.

☆ Once the data segment has begun, no comments may come within the data area.

☆ The individual fields of the data segment are delimited by assigning them to individual cells of the spreadsheet.

☆ All rows must contain the same number of filled cells. No cells - except the right-hand trailling cells - may be left empty. Any cell within the data block must contain numeric values. If no data is available for any particular cell, that cell must be filled with a dummy number, possibly 999, or 0, or -1. Select a dummy number that is easily identified (such as 999) so it will not be confused with real data.

☆ The left-most column (first colum) must contain a day reference, or be filled with a series of dummy numerals. No actual weather data may occupy column 1.

LOTUS™ 1-2-3 DATA FILE EXAMPLE

An extract of the first 10 lines of a weather data file is shown below.

```
% Douglas: Aug-28,1985..Dec-31,1987
%ID Amx Amn Smx Smn Rmm a/s h/b |8a <8a |2p <2p

240 275  55 250 125   0   0   0   0   0   0
241 270  55 245 125   0   0   0   0   0  50 31
242 275  60 235 130   0   0   0   0   0  50 36
243 270  60 250 120   0   0   0   0   0   0
244 275 100 265 145   0   0   0  50  90   0
245 290 115 260 150   0   0   0  50  90 100 3[illegible]

246 270  95 225 155  78   0   0   0   0 100 230
```

⇦ Comment line.
⇦ Additional comment line, identifies fields.
⇦ Blank line - mandatory
⇦ *First line of data segment. Fields are delimited by space characters. Each line represents data for a single day*

⇦ Last line of data segment..

LOTUS™ 1-2-3 DATA FILE FIELD SPECIFICATION

The data file is required to be not less than 10 (A···J) cells wide. Additional columns may exist, but they will not be processed by the interpreter.

Column(s)	Data description
A	Day identifier. Three digit number denoting the day of the year. January 1st is day number 001, December 31st is day number 365 (non-leap years) and day number 366 (leap years only).
B	Maximum air temperature. Three digit number indicating the maximum air temperature measured in °C, for that day. A decimal point is assumed to precede the last digit hence an entry "275" is interpreted as 27.5°C. A "-" indicates a negative temperature, below freezing.
C	Minimum air temperature. Three digit number indicating the minimum air temperature measured in °C, for that day. A decimal point is assumed to precede the last digit, hence an entry "95" is interpreted as 9.5°C. This value can also be entered as "095". A "-" indicates a negative temperature, below freezing.
D	Maximum soil temperature. Three digit number indicating the maximum soil temperature measured in °C, for that day. A decimal point is assumed to precede the last digit, hence an entry "275" is interpreted as 27.5°C. A "-" indicates a negative temperature, below freezing.
E	Minimum soil temperature. Three digit number indicating the minimum soil temperature measured in °C, for that day. A decimal point is assumed to precede the last digit, hence an entry "75" is interpreted as 7.5°C. A "-" indicates a negative temperature, below freezing.
F	Rainfall. Three digit number indicating the rainfall measured in millimetres, for that day. No decimal point is assumed. Negative numbers are meaningless, and will be interpreted as zero rainfall.
G	Recorded numbers of adult swarms. Three digit number indicating the number of adult swarms observed on that day. This data is not used by the simulation. Plotting these data might provide useful comparisons. The format and structure of this field is entirely user-specified, **HOWEVER, the field may not be left blank.** If no data is available for this field, the columns must be filled with zeros, or any other numeric values.

H Recorded numbers of hopper bands. Three digit number indicating the number of hopper bands observed on that day. This data is not used by the simulation. Plotting these data might provide useful comparisons. The format and structure of this field is entirely user-specified, **HOWEVER, the field may not be left blank.** If no data is available for this field, the columns must be filled with zeros, or any other numeric values

I Wind magnitude at 08h00. Three digit number indicating the wind speed magnitude measured in knots, at 08h00 on that day. (one knot = one nautical mile per hour) A decimal point is assumed to precede the last digit, hence an entry "75" is interpreted as 7.5knots. Negative speeds are meaningless but a "-" will be interpreted to mean that the wind direction is opposite to that specified in the "direction" field.

J Wind heading (direction) at 08h00. Three digit number indicating the wind heading measured in decimal degrees at 08h00, for that day. No decimal point is assumed. Angles are measured relative to true north, in a counter-clockwise direction. A "-" will be interpreted as a negative angle, which corresponds to a positive angle of the same magnitude but measured in a clockwise direction. Negative angles should be avoided, since values larger than 99 cannot be accommodated. Valid angles fall in the range 0···360° inclusive. Note that the wind heading is **the direction from which the wind blows**, not the direction to which it blows. Hence a wind heading of 0° implies the wind is northerly, or blowing towards the south.

K··· Ignored.

ADDING A NEW, PREVIOUSLY UNRECOGNISED STATION

You need to create a weather data file containing appropriate data for the new weather station. This is most easily achieved using a spreadsheet application such as Lotus 1-2-3, or QuattroPro, or a database application such as D-Base.

The simulation model can directly read the spreadsheet or Dbase file. If you want to use the ASCII data file format, it is necessary to export the data from the spreadsheet to an ASCII text file. Most spreadsheets allow this through the print-to-file option. You may need to specify that your printer is a "generic" or "text only" printer (even if your actual printer is not either of these!). Consult your spreadsheet manuals if you need help on generating ASCII text file output.

HINT: In order to get started, you might prefer to use an existing data file, which you simply copy and rename. In this way you will have a useable data file - with an appropriate name - although the data it contains will be inappropriate for the new site. Afterwards, once you are sure you have mastered the process of adding new sites to the simulation model, you can delete this file and replace it with appropriate data.

NOTE: Do not try to edit the Matlab script files loaddata.m, got_a.m, etc. while a simulation is running. To avoid getting a "Sharing Violation On Drive *N:*" error, you should close down Matlab entirely before trying to edit any script files.

STEP-BY-STEP EXAMPLE

Illustration of adding a new station is best done by way of example.

Suppose you wish to add a new weather station, situated at Richmond.

1) You need the latitude (south of the equator) and longitude (east of Greenwich) co-ordinates of the location. These are given as 31°25'S 23°56'E. Be careful to note whether your co-ordinate data is given in decimal-degrees, or degrees·minutes·seconds. The above data is supplied in DMS format, and needs to be converted to decimal degrees, which will be denoted with a subscript 10 ($_{10}$) to avoid confusion. Most scientific calculators (look for a "→**DEG**" key) can do this conversion with a few simple keystrokes. Alternatively, Windows' **Calculator** application can do this using the **INV DMS** function. Calculator can be found under the **Accessories** program group of any standard Windows set-up.
   31°25' → INV DMS → 31.4166_{10}°
   23°56' → INV DMS → 23.9333_{10}°

2) Choose an unambiguous station code for the new town or station. You cannot use a code that is already in use. The code is limited to two lowercase characters, one of which may be a blank space, or a numeral. Punctuation marks and extended ASCII characters are not valid. Let us use "rm" as the code for Richmond.

3) Create a weather data file for the town of Richmond; call the file *rmasrw.m*. or *rmasrw.wk1*, depending on the file type. If you have no data as yet, and wish to use any data as part of an experimental trial, you can copy one of the existing data files and use the copy. Remember that the data will be completely unrealistic. To create such a trial data file, issue the following command at the DOS command prompt: **copy dgasrw.* rmasrw.*** ↵. Or use Windows Filemanager to copy a file.
   This creates a file *rmasrw.m* (or *rmasrw.wk1*) that is identical to *dgasrw.m* (*rmasrw.wk1*) except for the name. We can imagine that the data in the new file applies to Richmond.

4) Use a text editor to open the Matlab script file named **loaddlg.m**. Locate the line beginning with the instruction "**textstring =** " shown below:

```
textstring                                                    =
'Douglas|G-Reinett|Middelburg|Pofadder';
```

   More names may exist than are shown in the above example. Edit this line to incorporate the name of the new station, thus:

```
textstring                                                    =
'Douglas|G-Reinet|Middelburg|Pofadder|Richmond';
```

   Make sure that the new name is added at the end of the list Do not alter the order of existing names. Do not change the punctuation. Use a pipe "|" symbol (the same keyboard key as the backwards slash "\") to

separate the new station name from the others in the list. You can use any name or nickname you want for the new station. "Rmond" or "Rchmnd" would have been equally acceptable.

Take note of the number of names preceding the new name. This number is needed later on. In the above example, there are FOUR names preceding Richmond, so Richmond is the FIFTH name in the list.

5) Locate the section in **got_a.m** that contains the instructions:

```
elseif val == 'number'
       val =  'code'
```

After the *last* of these instruction blocks just prior to the "end" statement, add these three lines:

```
elseif val == '5'
  val = 'rm';
```

Your **got_a.m** file will now look something like this:

```
val = get(hpop,'Value');
if   val == 1
     val = 'dg';
elseif val == 2
     val = 'gr';
elseif val == 3
     val = 'mb';
elseif val == 4
     val = 'pa';
elseif val == 5
     val = 'rm';
end
close(fignum);
clear fignum;
clear hpop Dlgname;
```

Notice that the two character code for Richmond "rm" that was chosen earlier has been used in the `val =` statement. Also, remember that Richmond was fifth in the list of station names? Hence it has been compared to 5 in the `elseif val ==` statement.

6) Locate the section on **loadata.m** that contains the instructions:

```
elseif aa == 'code'
%      filename;
       DATA = wklread('filename',-3,0);
       town = 'townname';   %%% any useful comments %%%
```

Add a section to read:

```
        elseif aa == 'rm'
%         rmasrw;
          DATA = wk1read('rmasrw',-3,0);
          town = 'Richmond';   %%% a new file %%%
```

7) Save the changes to **loadata.m**

9) Save the changes to **got_a.m**

10) Use your text editor to open the script file **locate.m**
Find the section containing the listings of latitude: and longitude:

```
douglas      = [29.07 23.75];
graaffreinet = [32.25 24.53];
middelburg   = [31.48 25.03];
pofadder     = [29.13 19.38];
```

To this section, add the latitude and longitude co-ordinates, measured in decimal degrees.
Latitude - south of the equator is specified first, followed by longitude - east of Greenwich.
Add the line:

```
richmond     = [31.42 23.93];
```

using one space between the co-ordinate pair, and a period "." as a decimal point.

11) Find the section in **locate.m** with the instructions:

```
elseif aa == 'code'
      aa = stationname;
```

To this section, add the following two lines:

```
elseif aa == 'rm'
      aa = richmond;
```

12) Repeat modifications outlined in (12) above, for the section containing the instructions:

```
elseif bb == 'code'
      bb = stationname;
```

adding the following two lines:

```
elseif bb == 'rm'
      bb = richmond;
```

13) Save the changes to **locate.m**

14) Restart Matlab and check if you can select the new station from the menus without any errors occurring.

NOTE: Be careful to follow the punctuation shown in the example.
Make sure the case of your variables and town names matched that of the names in the example. Most errors - if they occur - are likely to be caused by punctuation errors and using the wrong case. An identifier such as "Richmond" is not interchangeable with "richmond"!

Appendix F Decision rules - quick reference

GENERAL FORMAT OF THIS SECTION

Each section in this appendix is associated with a Matlab function. Matlab functions are contained in .M files (similar to script files). Each .M file contains program code for a single function, and the name of the .M file is suggestive of the function name. For example, the function NONDIAPAUSE is contained in a file name NONDIAP.M

A Matlab function is called by each lifecycle stage in the block diagram. That function makes decisions as to whether individuals or eggs may proceed to the next lifecycle stage or not. For example, considering the lifecycle stage *Non Diapause*, the function NONDIAPAUSE makes decisions as to whether eggs at the non-diapause stage of the lifecycle may proceed to subsequent stages, or remain in non-diapause, or die.

NOTE: "iff" is an acronym for 'if and only if'.

Each rule outline has the following format:

FUNCTION NAME (the identifier used by the Simulink block diagram to call this function)

filename	:	DOS name of disk file containing program code for this function.
input	:	local variable indicating how many individuals enter this lifecycle stage.
parameters	:	user-adjustable parameters affecting the conditions on which decisions are made.

☆ Descriptions of the decision rules follow each bullet.

RULES FOR TRANSITIONS BETWEEN LIFECYCLE STAGES

NONDIAPAUSE

filename	:	nondiap.m	
input	:	numberlaid	: sum of laid non-diapause eggs
parameters	:	DDthreshold	: required *days to induce hatching*
		catatrepsisafter	: required consecutive wet days
		catmaxdrydays	: death after consecutive dry days if passed catatrepsis
		waterloggedafter	: death after consecutive raining days
		quiescentafter	: required consecutive dry days
		fertility	: scaling factor of output
		predation	: proportion surviving

☆ Eggs hatch after accumulating 'DDthreshold' degree days, iff they have passed catatrepsis.
☆ Eggs die if they endure 'waterloggedafter' consecutive raining days.
☆ Eggs progress to catatrepsis if they experience 'catatrepsisafter' consecutive wet days.
☆ Catatrepsis eggs die if they experience more than 'catmaxdrydays' consecutive dry days.
☆ Eggs progress to QUIESCENT1 if they are still in NONDIAPAUSE after 'quiescentafter' days.

DIAPAUSE

filename	:	diapause.m	
input	:	numberlaid	: sum of laid diapause eggs
parameters	:	numdaysdiap	: max. allowed days in DIAPAUSE
	:	q1tod1ratio	: proportion of dry-laid eggs going to QUIESCENT1

☆ Eggs leave DIAPAUSE after 'numdaysdiap'.
☆ Eggs laid on a dry day progress to DIAPAUSE1 and QUIESCENT1.
☆ a fraction 'q1tod1ratio' of those eggs go to QUIESCENT1, (1-'q1tod1ratio') go to DIAPAUSE1.
☆ Eggs laid on a wet day progress to DIAPAUSE2.

DIAPAUSE1

filename	:	diapaus1.m	
input	:	numfromd	: eggs from DIAPAUSE
parameters	:	maxdrydays	: maximum dry days before death is caused
	:	maxwetdays	: max. consecutive wet days before progressing to TURGID2/DIAPAUSE2
	:	maxdaysd1	: max. days in DIAPAUSE1 before death occurs
	:	t1outputratio	: TURGID1:DIAPAUSE2 output split ratio

☆ Eggs die if they remain in DIAPAUSE1 longer than 'maxdaysd1'.
☆ Eggs die if they experience more than 'maxdrydays'
☆ Eggs experiencing more than 'maxwetdays' wet days progress to TURGID2 and/or DIAPAUSE2.
☆ Eggs leaving to TURGID2 and/or DIAPAUSE2 are split in the ratio TURGID2:DIAPAUSE2 = t1outputratio.

DIAPAUSE2

filename	:	diapaus2.m	
input	:	numberfromd1	: eggs from DIAPAUSE1
	:	numberfromd	: eggs from DIAPAUSE
parameters	:	numdaysd2	: max. days allowed in DIAPAUSE2

☆ Eggs progress to TURGID2 or to QUIESCENT2 after experiencing more than 'numdaysd2' in DIAPAUSE2.
☆ Eggs progress to TURGID2 if it is wet on the day of maturing (today).
☆ Eggs progress to QUIESCENT2 if it is dry on the day of maturing (today).

QUIESCENT1

filename	:	quiesnt1.m	
input	:	numberfromnd	: eggs from NONDIAPAUSE
	:	numberfromd	: eggs from DIAPAUSE
parameters	:	dehydrate	: max. dry days before death results
	:	maxdaysq1	: max. days allowed in QUIESCENT1
	:	maxwetdayst1	: consecutive wet days for eggs to proceed to TURGID1

☆ Eggs experiencing more than 'maxdaysq1' in QUIESCENT1 die.
☆ Eggs experiencing more than 'maxdrydays' consecutive dry days in QUIESCENT1 die.
☆ Eggs experiencing 'maxwetdayst1' or more consecutive wet days proceed to TURGID1.

QUIESCENT2

filename	:	quiesnt2.m	
input	:	numberfromt2	: eggs from TURGID2
	:	numberfromt1	: eggs from TURGID1
	:	numberfromd2	: eggs from DIAPAUSE2
parameters	:	dehydrate	: max. dry days before death re .s
	:	maxwetdayst2	: max. wet days for eggs to proceed to TURGID2
	:	maxdaysq2	: max. days allowed in QUIESCENT2

☆ Eggs experiencing more than 'maxdaysq2' in QUIESCENT2 die.
☆ Eggs experiencing more than 'maxdrydays' consecutive dry days in QUIESCENT2 die.
☆ Eggs experiencing 'maxwetdayst2' or more consecutive wet days proceed to TURGID2.

TURGID1

filename	:	turgid1.m	
input	:	numberfromq1	: eggs from QUIESCENT1
	:	numberfromd1	: eggs from DIAPAUSE1
parameters	:	DDthreshold	: °days that eggs must experience to hatch
	:	catatrepsisafter	: consecutive wet days required to reach catatrepsis
	:	catmaxdrydays	: consecutive dry days required for catatrepsis eggs to die
	:	waterloggedafter	: consecutive wet days required to cause death
	:	quiescentafter	: max. dry days allowed in TURGID1, forced to next stage
	:	fertility	: scaling factor for output
	:	predation	: survival rate

☆ Catatrepsis eggs that accumulate more than 'DDthreshold' °days proceed to hatch.
☆ Hatchlings are modified by scaling factors 'predation' and 'infertility'.
☆ Eggs experiencing more than 'waterloggedafter' consecutive rainy days die.
☆ Eggs experiencing 'catatrepsisafter' consecutive wet days proceed to catatrepsis.
☆ Catatrepsis eggs experiencing more than 'catmaxdrydays' consecutive dry days die.
☆ Eggs still in TURGID1 longer than 'quiescentafter' days proceed to QUIESCENT2.

TURGID2

filename	:	turgid2.m	
input	:	numberfromq2	: eggs from QUIESCENT2
	:	numberfromd2	: eggs from DIAPAUSE2
parameters	:	DDthreshold	: °days that eggs must experience to hatch
	:	catatrepsisafter	: consecutive wet days required to reach catatrepsis
	:	catmaxdrydays	: consecutive dry days required for catatrepsis eggs to die
	:	waterloggedafter	: consecutive wet days required to cause death
	:	quiescentafter	: max. dry days allowed in TURGID2, forced to next stage
	:	fertility	: scaling factor for output
	:	predation	: survival rate

☆ Catatrepsis eggs that accumulate more than 'DDthreshold' °days proceed to hatch.
☆ Hatchlings are modified by scaling factors 'predation' and 'infertility'.
☆ Eggs experiencing more than 'waterloggedafter' consecutive rainy days die.
☆ Eggs experiencing 'catatrepsisafter' consecutive wet days proceed to catatrepsis.
☆ Catatrepsis eggs experiencing more than 'catmaxdrydays' consecutive dry days die.
☆ Eggs still in TURGID2 longer than 'quiescentafter' days proceed to QUIESCENT2.

INSTARS

filename	:	instars.m	
input	:	numberhatching	: hatchlings from TURGID1 & TURGID2 & NONDIAPAUSE
parameters	:	DDthreshold	: °days that hatchlings must experience to advance to maturing adults
	:	survival	: survival rate

☆ Hatchlings that accumulate more than 'DDthreshold' °days proceed to next stage.
☆ Hatchlings are modified by scaling factor 'survival'.

MATURINGADULT

filename	:	matad.m	
input	:	numbermoulting	: young adults from INSTARS
parameters	:	DDthreshold	: °days that maturing adults must experience to advance to adults
	:	survival	: survival rate
	:	sexratio	: proportion of adults that are female

☆ Maturing adults that accumulate more than 'DDthreshold' °days proceed to next stage.
☆ Maturing adults leaving are modified by scaling factor 'survival'.
☆ Surviving maturing adults are split into female:male in the ratio 'sexratio':(1–'sexratio').

ADULTMALES

filename	:	adltmale.m	
input	:	numfrommatadm	: male adults from MATURINGADULT
parameters	:	adultmalelifespan	: max. days in ADULTMALES before death

☆ Adult males die after experiencing 'adultmalelifespan' days in ADULTMALES.

LAY1

filename	:	lay1.m	
input	:	numfrommatadf	: female adults from MATURINGADULT
parameters	:	phaseratio	: proportion of gregarious:solitary females
	:	layinterval	: maximum days in LAY1
	:	survival	: proportion of survivors
	:	gregnondiapratio	: proportion of non-diapause eggs (cf., diapause eggs) from gregarious females
	:	soldiapratio	: proportion of diapause eggs (cf. non-diapause eggs) from solitary females
	:	eggspergregpacket	: number of eggs in packets laid by gregarious females
	:	eggspersolpacket	: number of eggs per packet laid by solitary females
	:	flightlempthreshold	: minimum temperature to enable flight
	:	minflightage	: number of days in this laying stage before which flight cannot occur

☆ Each solitary female lays one egg packet upon entering this stage.
☆ Each gregarious female lays one egg packet upon entering this stage.
☆ Diapause eggs are tallied.
☆ Non-diapause eggs are tallied.
☆ Females proceed to next laying stage after remaining in LAY1 for "layinterval" days.
☆ Proceeding female numbers modified by "survival" factor.

LAY2

filename	:	lay2.m	
input	:	numgregfromlay1	: gregarious female adults from LAY1
	:	numsolfromlay1	: gregarious female adults from LAY1
parameters	:	layinterval	: maximum days in LAY2
	:	survival	: proportion of survivors
	:	gregnondiapratio	: proportion of non-diapause eggs (cf., diapause eggs) from gregarious females
	:	soldiapratio	: proportion of diapause eggs (cf. non-diapause eggs) from solitary females
	:	eggspergregpacket	: number of eggs in packets laid by gregarious females
	:	eggspersolpacket	: number of eggs per packet laid by solitary females
	:	flightlempthreshold	: minimum temperature to enable flight
	:	minflightage	: number of days in this laying stage before which flight cannot occur

☆ Each solitary female lays one egg packet upon entering this stage.
☆ Each gregarious female lays one egg packet upon entering this stage.
☆ Diapause eggs are tallied.
☆ Non-diapause eggs are tallied.
☆ Females proceed to next laying stage after remaining in LAY2 for "layinterval" days.
☆ Proceeding female numbers modified by "mortality" factor.

LAY3

filename	:	lay3.m	
input	:	numgregfromlay2	: gregarious female adults from LAY2
	:	numsolfromlay2	: gregarious female adults from LAY2
parameters	:	layinterval	: maximum days in LAY3
	:	survival	: proportion of survivors
	:	gregnondiapratio	: proportion of non-diapause eggs (cf., diapause eggs) from gregarious females
	:	soldiapratio	: proportion of diapause eggs (cf. non-diapause eggs) from solitary females
	:	eggspergregpacket	: number of eggs in packets laid by gregarious females
	:	eggspersolpacket	: number of eggs per packet laid by solitary females
	:	flighttempthreshold	: minimum temperature to enable flight
	:	minlightage	: number of days in this laying stage before which flight cannot occur

☆ Each solitary female lays one egg packet upon entering this stage.
☆ Each gregarious female lays one egg packet upon entering this stage.
☆ Diapause eggs are tallied.
☆ Non-diapause eggs are tallied.
☆ Females proceed to next laying stage after remaining in LAY3 for "layinterval' days.
☆ Proceeding female numbers modified by 'survival' factor.

LAY4

filename	:	lay4.m	
input	:	numgregfromlay1	: gregarious female adults from LAY3
	:	numsolfromlay1	: gregarious female adults from LAY3
parameters	:	lifespan	: maximum days in LAY4
	:	gregnondiapratio	: proportion of non-diapause eggs (cf., diapause eggs) from gregarious females
	:	soldiapratio	: proportion of diapause eggs (cf. non-diapause eggs) from solitary females
	:	eggspergregpacket	: number of eggs in packets laid by gregarious females
	:	eggspersolpacket	: number of eggs per packet laid by solitary females
	:	flighttempthreshold	: minimum temperature to enable flight
	:	minflightage	: number of days in this laying stage before which flight cannot occur

☆ Each solitary female lays one egg packet upon entering this stage.
☆ Each gregarious female lays one egg packet upon entering this stage.
☆ Diapause eggs are tallied.
☆ Non-diapause eggs are tallied.
☆ Females die after remaining in LAY2 for "layinterval' days.

RULES FOR MIGRATION BETWEEN NODES

ATOB

filename	: atob.m	
input	: lay1	: insects at LAY1 developmental stage leaving station A
	: lay2	: insects at LAY2 developmental stage leaving station A
	: lay3	: insects at LAY3 developmental stage leaving station A
	: lay4	: insects at LAY4 developmental stage leaving station A
parameters	:	:

☆ Each swarm (comprising individuals from LAY1...LAY4 individuals) is scaled by a numerical factor derived from WINDINDEX column 7, corresponding to CURRENTDAY's wind data.

BTOA

filename	: btoa.m	
input	: lay1	: insects at LAY1 developmental stage leaving station B
	: lay2	: insects at LAY2 developmental stage leaving station B
	: lay3	: insects at LAY3 developmental stage leaving station B
	: lay4	: insects at LAY4 developmental stage leaving station B
parameters	:	:

☆ Each swarm (comprising individuals from LAY1...LAY4 individuals) is scaled by a numerical factor derived from WINDINDEX column 8, corresponding to CURRENTDAY's wind data.

Appendix G Population variables - quick reference

GLOBAL AND LOCAL VARIABLES

A variable is a named entity that may take on a numeric value. For example, the simulation model references a variable called CURRENTDAY. CURRENTDAY may take on an integer value anywhere in the range "0" to positive infinity, the value representing the number of days that have elapsed since the simulation began. Not all variables are confined to be positive, or integers. Neither are they confined to be scalar types; they may represent two-dimensional arrays of numbers.

There exist global variables and local variables. Global variables are available to be read and modified by all the functions used in the simulation model. Local variables are only visible to those functions in which they are defined. Functions cannot read or modify the local variables of any other function. CURRENTDAY is a global variable.

There is no need for the user to be familiar with the local variables.

All population variables are global variables. This is necessary in order that the se variables be available to other functions such as the plotting routines.

GENERAL STRUCTURE OF POPULATION VARIABLES

In any variable, the width (number of columns) of the array is representative of the number of days that any group of individuals has been in that stage of development. Groups of individuals become progressively older from left-to-right.

New individuals entering any stage are always entered into column 1 (left most column), the existing columns all being shifted to the right by one. The existing individuals in a stage are advanced one column to the right each day regardless of whether any individuals are entering that stage or not. In the latter case, column 1 is filled with zeros for that day.

Individuals that have experienced sufficiently long durations of necessary conditions to advance to the next stage of development are deleted from the right-hand-side of the array. Trailing zeros on the right-hand-side of any array are always truncated.

The precise number of individuals that have been in any stage for j days will be found in the jth column of that array. Numbers of individuals are stored as floating-point reals; fractions of an individual or of an egg are thus allowed.

POPULATION VARIABLES

NONDIAPAUSE
A 3xn matrix

1,1	1,2	1,3	1,4	...	1,n
2,1	2,2	2,3	2,4	...	2,n
3,1	3,2	3,3	3,4	...	3,n

Entries in the jth column have been in NONDIAPAUSE for j days.
(1,j) = quantity of eggs that have been in NONDIAPAUSE for j days.
(2,j) = accumulated degree-days for eggs in NONDIAPAUSE for j days.
(3,j) = 1 or 0, j=0 iff eggs have reached catatrepsis on day j.

DIAPAUSE
A 2xn matrix

1,1	1,2	1,3	1,4	...	1,n
2,1	2,2	2,3	2,4	...	2,n

Entries in the jth column have been in DIAPAUSE for j days.
(1,j) = quantity of eggs laid in wet conditions, in DIAPAUSE for j days.
(2,j) = quantity of eggs laid in dry conditions, in DIAPAUSE for j days.

DIAPAUSE1
A 1xn matrix

1,1	1,2	1,3	1,4	...	1,n

Entries in the jth column have been in DIAPAUSE1 for j days.
(1,j) = quantity of eggs that have been in DIAPAUSE1 for j days.

DIAPAUSE2
A 1xn matrix

1,1	1,2	1,3	1,4	...	1,n

Entries in the jth column have been in DIAPAUSE2 for j days.
(1,j) = quantity of eggs that have been in DIAPAUSE2 for j days.

QUIESCENT1
A 1xn matrix

1,1	1,2	1,3	1,4	...	1,n

Entries in the jth column have been in QUIESCENT1 for j days.
(1,j) = quantity of eggs that have been in QUIESCENT1 for j days.

QUIESCENT2
A 1xn matrix

1,1	1,2	1,3	1,4	...	1,n

Entries in the *j*th column have been in QUIESCENT2 for *j* days.
(1,j) = quantity of eggs that have been in QUIESCENT2 for j days.

TURGID1
A 3xn matrix

1,1	1,2	1,3	1,4	...	1,n
2,1	2,2	2,3	2,4	...	2,n
3,1	3,2	3,3	3,4	...	3,n

Entries in the *j*th column have been in TURGID1 for *j* days.
(1,j) = quantity of eggs that have been in TURGID1 for j days.
(2,j) = acceumulated degree-days for eggs in TURGID1 for j days.
(3,j) = 1 or 0, j=0 iff eggs have reached catatrepsis on day j.

TURGID2
A 3xn matrix

1,1	1,2	1,3	1,4	...	1,n
2,1	2,2	2,3	2,4	...	2,n
3,1	3,2	3,3	3,4	...	3,n

Entries in the *j*th column have been in TURGID2 for *j* days.
(1,j) = quantity of eggs that have been in TURGID2 for j days.
(2,j) = accumulated degree-days for eggs in TURGID2 for j days.
(3,j) = 1 or 0, j=0 iff eggs have reached catatrepsis on day j.

INSTARS
A 2xn matrix

1,1	1,2	1,3	1,4	...	1,n
2,1	2,2	2,3	2,4	...	2,n

Entries in the *j*th column have been in INSTARS for *j* days.
(1,j) = quantity of hoppers that have been in INSTARS for j days.
(2,j) = accumulated degree-days for hoppers in INSTARS for j days.

MATURINGADULT
A 2xn matrix

1,1	1,2	1,3	1,4	...	1,n
2,1	2,2	2,3	2,4	...	2,n

Entries in the jth column have been in MATURINGADULT for *j* days.
(1,j) = quantity of insects that have been in MATURINGADULT for j days.
(2,j) = accumulated degree-days for insects in MATURINGADULT for j days.

ADULTMALES
A 1xn matrix

1,1	1,2	1,3	1,4	..	1,n

Entries in the jth column have been in ADULTMALES for *j* days.
(1,j) = quantity of adult males that have been in ADULTMALES for j days.

LAY1
A 2xn matrix

1,1	1,2	1,3	1,4	...	1,n
2,1	2,2	2,3	2,4	...	2,n

Entries in the jth column have been in LAY1 for *j* days.
(1,j) = quantity of gregarious females that have been in LAY1 for j days.
(2,j) = quantity of solitary females that have been in LAY1 for j days.

LAY2
A 2xn matrix

1,1	1,2	1,3	1,4	...	1,n
2,1	2,2	2,3	2,4	...	2,n

Entries in the jth column have been in LAY2 for *j* days.
(1,j) = quantity of gregarious females that have been in LAY2 for j days.
(2,j) = quantity of solitary females that have been in LAY1 for j days.

LAY3
A 2xn matrix

1,1	1,2	1,3	1,4	...	1,n
2,1	2,2	2,3	2,4	...	2,n

Entries in the jth column have been in LAY3 for *j* days.
(1,j) = quantity of gregarious females that have been in LAY3 for j days.
(2,j) = quantity of solitary females that have been in LAY3 for j days.

LAY4
A 2xn matrix

1,1	1,2	1,3	1,4	...	1,n
2,1	2,2	2,3	2,4	...	2,n

Entries in the jth column have been in LAY4 for j days.
(1,j) = quantity of gregarious females that have been in LAY4 for j days.
(2,j) = quantity of solitary females that have been in LAY4 for j days.

Appendix H Lifecycle model and simulation parameters in use during test simulations.

NOTE: These parameters will be better understood if referred to in conjunction with the decision rules.

NOTE: Remember to save your changes if you want new values to be available during subsequent sessions.

The parameter names listed below, as used in the block diagram are similar to, but not absolutely identical to the corresponding source code variable names. The source code variable names are those listed in the section describing the decision rules. The block diagram variable names are completely arbitrary - the simulation never actually reads these names, they are there only for ease of understanding by the user. These block diagram variable names may be altered at will, but you should not rename them to something totally unrelated to the source code variable since it will then be difficult to keep track of their function.

USER-ADJUSTABLE LIFECYCLE PARAMETERS THAT CAN BE ACCESSED FROM THE BLOCK DIAGRAM

PARENT BLOCK	PARAMETER IDENTIFIER	DESCRIPTION [recommended value(s) shown in square brackets]
adultmales	adultmalelifespan	number of days a mature male insect exists before being deleted from the population record. [40]
combine	(none)	
demux	(none)	
diapause	numdaysdiap	number of days that eggs remain in diapause before progressing to diapause1 or quiescent1. [12]
	q1 to d1 ratio	proportion of individuals leaving diapause that will go to quiescent1 instead of diapause1. [0.95]
diapause 1	maxdrydays	maximum number of consecutive dry days that eggs can experience before being deleted from population record, [300]
	maxdaysd1	maximum number of days that eggs can remain in diapause1. Eggs not progressing to subsequent stages after "max days in diap1" days will be deleted from the population record. [1095]
	maxwetday	maximum number of consecutive wet days eggs may experience before progressing to one of turgid1 or diapause2. [2]
	tt outputratio	proportion of hydrated eggs that progress to turgid1. The complement progresses to diapause2. [0.5]
diapause 2	numdaysd2	number of days that eggs remain in diapause2 before progressing to turgid2 or quiescent2. [45]
eggsum	(none)	

Instars	DDthreshold	required accumulated degree-days for hoppers to progress to the subsequent stage. [8]
	survival	proportion of hoppers that progress to subsequent stage. The complement is deleted from the population record. [0.5]
lay1 (only)	phase ratio	proportion of laying females that are gregarious. The complement is solitary. [0.5]
lay1 ··· lay4	layinterval	number of days that individuals remain in any laying stage before proceeding to the subsequent stage. [7]
	survival rate	proportion of adult females that progress to subsequent stage. The complement is deleted from the population record. [0.5]
	gregnondiapauseratio	proportion of non-diapause eggs laid by gregarious females. The complement is diapause eggs. [0.5]
	soldiapauseratio	proportion of diapause eggs laid by solitary females. The complement is non-diapause eggs. [0.5]
	gregeggsperpacket	number of eggs per gregarious egg packet. [45]
	soleggsperpacket	number of eggs per solitary egg packet. [80]
	flight temp threshold	temperature below which migratory flight is inhibited. [20]
	min flight age	number of days that laying female has to have been in any laying stage before migratory flight is allowed. [3]
lay4 (only)	lifespan	number of days a lay4 stage female remains in lay4 before being deleted from the population record. [7]
maturing adult	DDThreshold	necessary accumulated degree-days for a maturing adult to progress to subsequent stage. [80]
	survival	proportion of maturing adults that progress to subsequent stage. The complement is deleted from the population record. [0.5]
	sexratio	proportion of individuals that is female. The complement is male. [0.5]
memory	(none)	
mux	(none)	
non diapause	DDThreshold	required accumulated degree-days for diapause eggs to proceed to subsequent stage, if and only if they have passed catatrepsis. [30]
	catatrepsis after	required consecutive wet days for eggs to reach catatrepsis stage of development. [3]
	waterlogged after	required consecutive wet days to cause death. Eggs are deleted from the population record. [16]
	quiescent after	required consecutive dry days for eggs to progress to quiescent stage. [12]

	cat max dry days	required consecutive dry days for catatrepsis eggs to be deleted from population record. [21]
	fertility	proportion of eggs that are allowed to hatch. The complement is deleted from the population record. [0.75]
	predator survival	proportion of eggs that are allowed to progress to subsequent stages. The complement is deleted from the population record. [0.75]
quiescent 1	dehydrate	number of consecutive dry days that eggs may experience before being deleted from the population record. [300]
	maxdaysq1	number of consecutive days that eggs may remain in quiescent1 before being deleted from population record. [1095]
	maxwetdayst1	number of consecutive wet days that will cause eggs to progress to turgid1. [2]
quiescent 2	dehydrate	number of consecutive dry days that eggs may experience before being deleted from the population record. [300]
	maxdaysq2	number of consecutive days that eggs may remain in quiescent1 before being deleted from population record. [1095]
	maxwetdayst2	number of consecutive wet days that will cause eggs to progress to turgid1. [2]
turgid 1	DDThreshold	number of accumulated degree days necessary for catatrepsis eggs to hatch. [45]
	catatrepsisafter	number of consecutive wet days necessary for eggs to pass catatrepsis stage of development. [3]
	cat maxdrydays	number of consecutive dry days catatrepsis eggs may experience before being deleted from the population record. [3]
	waterloggedafter	number of consecutive wet days that eggs may experience before being deleted from the population record. [16]
	quiescentafter	maximum number of days that eggs may remain in turgid1. Turgid1 eggs will proceed to quiescent2 after quiescentafter days. [40]
	fertilit	proportion of eggs that are allowed to hatch. The complement is deleted from the population record. [0.75]
	predator survival	proportion of eggs that are allowed to progress to subsequent stages. The complement is deleted from the population record. [0.75]
turgid 2	DDThreshold	number of accumulated degree days necessary for catatrepsis eggs to hatch. [30]

catatrepsisafter	number of consecutive wet days necessary for eggs to pass catatrepsis stage of development. [3]
catmaxdrydays	number of consecutive dry days catatrepsis eggs may experience before being deleted from the population record. [155]
waterloggedafter	number of consecutive wet days that eggs may experience before being deleted from the population record. 14]
quiescentafter	maximum number of days that eggs may remain in turgid1. Turgid2 eggs will proceed to quiescent2 after *quiescentafter* days. [900]
fertility	proportion of eggs that are allowed to hatch. The complement is deleted from the population record. [0.75]
predator survival	proportion of eggs that are allowed to progress to subsequent stages. The complement is deleted from the population record. [0.75]

USER-ADJUSTABLE LIFECYCLE PARAMETERS ACCESSIBLE ONLY IN THE SOURCE CODE FILES

NOTE: The parameter identifiers (the names of the variables) are case sensitive. Be careful when you edit these variables that you do not inadvertently change the spelling of the identifiers.

PARENT M-FILE	PARAMETER IDENTIFIER	DESCRIPTION [recommended value(s) shown in square brackets]
lcs_.m	ADULTMALES_	Sets the initial population of individuals in this lifecycle stage for the site corresponding to "_"
	LAY1_	Sets the initial population of individuals in this lifecycle stage for the site corresponding to "_"
	LAY2_	Sets the initial population of individuals in this lifecycle stage for the site corresponding to "_"
	LAY3_	Sets the initial population of individuals in this lifecycle stage for the site corresponding to "_"
	LAY4_	Sets the initial population of individuals in this lifecycle stage for the site corresponding to "_"
	DIAPAUSE_	Sets the initial population of individuals in this lifecycle stage for the site corresponding to "_"
	DIAPAUSE1_	Sets the initial population of individuals in this lifecycle stage for the site corresponding to "_"
	DIAPAUSE2_	Sets the initial population of individuals in this lifecycle stage for the site corresponding to "_"
	NONDIAPAUSE_	Sets the initial population of individuals in this lifecycle stage for the site corresponding to "_"
	TURGID1_	Sets the initial population of individuals in this lifecycle stage for the site corresponding to "_"

	TURGID2_	Sets the initial population of individuals in this lifecycle stage for the site corresponding to "_"
	QUIESCENT1_	Sets the initial population of individuals in this lifecycle stage for the site corresponding to "_"
	QUIESCENT2_	Sets the initial population of individuals in this lifecycle stage for the site corresponding to "_"
	INSTARS_	Sets the initial population of individuals in this lifecycle stage for the site corresponding to "_"
	MATURINGADULT_	Sets the initial population of individuals in this lifecycle stage for the site corresponding to "_"
precalc_.m	class2	defines the upper boundary of this category of rainfall. Rainfall between 0* and class2 mm is considered to be a rain fall of class 2. Rainfall = 0mm is assigned to class 1. [6]
	class3	defines the upper boundary of this category of rainfall. Rainfall between class2 and class3 mm is considered to be a rain fall of class 3. [10]
	class4	defines the upper boundary of this category of rainfall. Rainfall between class3 and class4 mm is considered to be a rain fall of class 4. [15]
	class5	defines the upper boundary of this category of rainfall. Rainfall between class4 and class5 mm is considered to be a rain fall of class 5. Rainfall above class5 mm is assigned to class 6. [25]
	lowerthreshold	temperature threshold for development. Degree-day contribution of any day is calculated for all temperature above lowerthreshold °C. If the day temperature remains below lowerthreshold °C, that day will have a degree-day contribution of zero. [20]
windcalc.m	Tthresh	temperature (°C) below which any flight is inhibited. Used to determine the fraction of the day that is conducive to flight. [18]
	maxhours	The maximum allowable flight period. If the day temperature remains above Tthresh °C for more than maxhours, the flight time will be truncated to be equal to maxhours. [20]
	beta	The coefficient of the exponential logarithm used to determine the spread (width) of the Gaussian probability function associated with wind direction matching relative bearings of stations. [100]
	bee	The coefficient of the exponential logarithm used to determine the spread (width) of the Gaussian probability function associated with flight distance matching interstation distance. [100]

Author:Tilch, David.
Name of thesis:Dynamic simulation modelling of locust outbreaks.

PUBLISHER:
University of the Witwatersrand, Johannesburg
©2015

LEGALNOTICES:

Copyright Notice: All materials on the University of the Witwatersrand, Johannesburg Library website are protected by South African copyright law and may not be distributed, transmitted, displayed or otherwise published in any format, without the prior written permission of the copyright owner.

Disclaimer and Terms of Use: Provided that you maintain all copyright and other notices contained therein, you may download material (one machine readable copy and one print copy per page)for your personal and/or educational non-commercial use only.

The University of the Witwatersrand, Johannesburg, is not responsible for any errors or omissions and excludes any and all liability for any errors in or omissions from the information on the Library website.