

# DYMEX

## Model Simulator

VERSION 3



## User's Guide

G.F. Maywald, W. Bottomley and R.W. Sutherst



Copyright © 2007 Hearne Scientific Software

## Hearne Scientific Software End-User Licence Agreement

This is a legal agreement between you (either an individual or an entity) and Hearne Scientific Software Pty Ltd. By opening the sealed Hearne Scientific Software package (herein SOFTWARE) and/or by using the SOFTWARE, you agree to be bound by the terms of this agreement. If you do not agree to the terms of this agreement, promptly return the SOFTWARE and accompanying items (including printed materials or other containers) to the place you obtained them for a full refund within thirty (30) days of the purchase date.

1. **Grant of Licence:** Hearne Scientific Software grants you a non-exclusive licence to use the SOFTWARE in accordance with the following terms.
  - a. **Single-User Licence:** If you purchased a single-user licence, Hearne Scientific Software allows one (1) designated individual, and only that individual, the right to install the software on one (1) home, one (1) work, and one (1) portable computer. The designated individual agrees that no more than one (1) installation will be in use at any given time.
  - b. **Group Licence-Limited Installation:** If you purchased a multi-user licence, you may install and use the SOFTWARE on individual computer workstations within your organisation up to the number of users specified in the purchase order. This type of licence is not portable outside your organisation (i.e. The SOFTWARE can not be installed on a computer at home or on a portable computer unless your organisation purchases a maintenance plan at the time of purchasing the group licence).
2. **Backup Copies:** Hearne Scientific Software grants you the right to make one (1) archival copy of the SOFTWARE for the sole purpose of backing up the SOFTWARE and protecting your investment from loss.
3. **Transfer:** Hearne Scientific Software further grants you the right to transfer this licence and the SOFTWARE to another party provided the following transfer conditions are met.
  - a. The other party accepts all terms of this agreement.
  - b. All copies of the SOFTWARE are transferred and you discontinue use of the SOFTWARE after transferring.
  - c. Hearne Scientific Software is promptly notified of the name and address of the other party and the serial number of the SOFTWARE.
  - d. Hearne Scientific Software is not required to supply new media.
4. **Term and Termination:** Failure to comply with any of these terms will terminate this agreement and your right to use the SOFTWARE. You may also choose to terminate the agreement at any time. Upon termination of this agreement, you must immediately destroy the SOFTWARE and all copies of it.
5. **Copyright:** The SOFTWARE (including any images, applets, photographs, animations, video, audio, music and text incorporated into the SOFTWARE) is owned by CSIRO and Hearne Scientific Software and is protected by Australian copyright laws and international treaty provisions. You agree not to modify, adapt, translate, reverse engineer, decompile, or disassemble the Software. You must treat the SOFTWARE like any other copyrighted material (e.g. a book or musical recording) except that you may either:
  - a. Make one copy of the SOFTWARE solely for backup or archival purposes; or
  - b. Transfer the SOFTWARE to a single hard disk provided you keep the original solely for backup or archival purposes.
  - c. You may only copy the printed materials accompanying the SOFTWARE for your own internal use.
6. **Limited Warranty:** Hearne Scientific Software warrants that the SOFTWARE will perform substantially in accordance with the accompanying printed materials for a period of thirty (30) days from the date of purchase. Any implied warranties on the SOFTWARE are limited to thirty (30) days. Some states/provinces do not allow limitations on duration of an implied warranty, so the above limitation may not apply to you.
7. **Customer Remedies:** Hearne Scientific Software's entire liability and your exclusive remedy shall be, at Hearne Scientific Software's option:
  - a. Return of the price paid; or
  - b. Repair or replacement of the SOFTWARE that does not meet Hearne Scientific Software's limited warranty and that is returned to Hearne Scientific Software with a copy of your receipt. This limited warranty is void if failure of the SOFTWARE or hardware has resulted from accident, abuse, or improper application. Any replacement of SOFTWARE will be warranted for a further thirty (30) days.
8. **No Other Warranties:** To the maximum extent permitted by applicable law, Hearne Scientific Software disclaims all other warranties, expressed or implied, including but not limited to implied warranties of merchantability and fitness for a particular purpose, with respect to the SOFTWARE and the accompanying written materials. You may have others, which vary from state to province to province.
9. **NO LIABILITY FOR CONSEQUENTIAL DAMAGES:** To the maximum extent permitted by applicable law, in no event shall Hearne Scientific Software or its suppliers be liable for any special, incidental, indirect, or consequential damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or any other pecuniary loss) arising out of the use of or inability to use the SOFTWARE product, even if Hearne Scientific Software has been advised of the possibility of such damages.
10. **Governing Law.** This Agreement will be governed by the laws in force in the State of Victoria, Australia.

Should you have any questions concerning this agreement please contact Hearne Scientific Software.

Hearne Scientific Software Pty Ltd  
Level 6, 552 Lonsdale St  
Melbourne 3000, Australia  
Ph +61 3 9602 5088  
Fx +61 3 9602 5050  
www.hearne.com.au

# Table of Contents

<b>1 INTRODUCTION.....</b>	<b>1</b>
1.1 WHAT IS THE SIMULATOR? .....	2
1.2 DIFFERENCES BETWEEN VERSIONS 1 AND 2 .....	2
1.3 DIFFERENCES BETWEEN VERSIONS 2 AND 3 .....	3
1.4 MINIMUM REQUIREMENTS TO RUN DYMEX .....	4
1.5 INSTALLING AND RUNNING DYMEX .....	4
<b>2 AN OVERVIEW OF DYMEX MODELS .....</b>	<b>4</b>
2.1 VARIABLES .....	4
2.2 MODULES .....	5
2.3 FUNCTIONS AND PARAMETERS .....	7
2.4 PROCESSES .....	8
2.5 SUB-POPULATIONS.....	8
2.5.1 Genetic sub-populations.....	9
2.5.2 Spatial sub-populations.....	10
2.6 DYMEX MODEL FILES .....	10
<b>3 OPENING A DYMEX MODEL.....</b>	<b>11</b>
3.1 ERRORS DURING MODEL LOADING .....	14
3.2 THE MODEL COMPONENTS WINDOW.....	15
3.3 MENU COMMANDS .....	17
3.4 MODEL DESCRIPTION .....	18
3.5 MODEL SUB-POPULATION STRUCTURE .....	18
3.6 PRINTING THE MODEL .....	19
3.7 MODULES .....	21
3.7.1 Module Information.....	21
3.7.2 Initializing Modules.....	22
<b>4 THE TIMER MODULE.....</b>	<b>23</b>
4.1 TIMER INITIALIZATION .....	23
<b>5 THE LIFECYCLE MODULE.....</b>	<b>24</b>
5.1 COHORTS AND COHORT PROPERTIES .....	25
5.2 LIFESTAGE PROCESSES AND THEIR CHARACTERISTICS .....	29
5.2.1 Mortality.....	30
5.2.2 Development.....	31

5.2.3 Dispersal Timing.....	31
5.2.4 Dispersal .....	31
5.2.5 Genetic Mixing.....	32
5.2.6 Stage Transfer .....	32
5.2.7 Fecundity and Progeny Production .....	33
5.3 DUMMY COHORTS .....	33
5.4 LIFECYCLE FACTORS .....	33
5.5 LIFECYCLE INITIALIZATION .....	34
5.6 THE LIFECYCLE DIAGRAM.....	38
5.7 THE LIFESTAGE WINDOW .....	40
5.8 EXAMINING THE COHORT PROPERTIES .....	41
5.9 COHORT REMOVAL.....	42
5.10 LIFECYCLE MODULE RUN OPTIONS .....	44
<b>6 THE DATAFILE AND METBASE MODULES .....</b>	<b>45</b>
6.1 METBASE & DATA FILE INITIALIZATION .....	45
6.1.1 Importing Spreadsheet Files into DYMEX.....	46
6.1.2 Initializing Data File Modules (Basic Options).....	46
6.2 INITIALIZING DATA FILE MODULES (MORE OPTIONS).....	49
<b>7 THE METMANAGER MODULE.....</b>	<b>52</b>
7.1 INITIALIZING THE METMANAGER MODULE.....	52
<b>8 THE CIRCADIAN MODULE.....</b>	<b>53</b>
8.1 INITIALIZING THE CIRCADIAN MODULE.....	55
<b>9 THE QUERYUSER AND QUERYFILE MODULES .....</b>	<b>56</b>
9.1 INITIALIZING THE QUERYUSER MODULE.....	56
9.2 INITIALISING THE QUERYUSER/DISCRETE MODULE .....	57
9.3 INITIALISING THE QUERYFILE MODULE .....	58
<b>10 THE EVENT MODULE .....</b>	<b>60</b>
10.1 INITIALISING THE EVENT MODULE .....	62
10.1.1 Setting a Date triggered event.....	63
10.1.2 Setting a Threshold triggered event .....	64
<b>11 THE EXPRESSION MODULE.....</b>	<b>66</b>
<b>12 THE FUNCTION MODULE.....</b>	<b>67</b>
<b>13 THE RUNNING MEAN MODULE.....</b>	<b>67</b>

<b>14 THE DEGREEDAY MODULE.....</b>	<b>67</b>
<b>15 THE EQUATION AND COUNTER MODULES .....</b>	<b>68</b>
<b>16 THE STORAGE MODULE.....</b>	<b>69</b>
16.1 INITIALISING THE STORAGE MODULE .....	70
<b>17 THE SOIL MOISTURE MODULE.....</b>	<b>71</b>
17.1 INITIALISING THE SOIL MOISTURE MODULE .....	71
<b>18 THE EVAPORATION MODULE .....</b>	<b>72</b>
<b>19 THE DAYLENGTH MODULE .....</b>	<b>72</b>
<b>20 THE CLIMATE CHANGE SCENARIO MODULE .....</b>	<b>73</b>
<b>21 THE DEMESPLITTER MODULE .....</b>	<b>73</b>
<b>22 THE DEMESTATISTICS MODULE.....</b>	<b>74</b>
<b>23 THE SUMMARYMANAGER MODULE.....</b>	<b>74</b>
<b>24 MODIFYING PARAMETERS .....</b>	<b>76</b>
24.1 THE PARAMETER “TREE” WINDOW .....	77
24.2 THE PARAMETER “GRID” .....	81
<b>25 SIMULATOR PREFERENCES .....</b>	<b>83</b>
25.1 OPERATING PREFERENCES .....	83
25.2 MODEL TRACE OPTIONS .....	85
25.3 CURRENT MODEL OPTIONS .....	86
25.4 THE HINTS WINDOW.....	89
<b>26 RUNNING THE MODEL.....</b>	<b>90</b>
26.1 RUNNING TO EQUILIBRIUM .....	90
26.1.1 <i>Defining the Equilibrium</i> .....	91
26.2 THE RUN WINDOW .....	92
26.3 RUN SUMMARIES .....	93
<b>27 RUN SEQUENCES.....</b>	<b>94</b>
27.1 EVENT RUN SEQUENCES .....	96
27.2 DATAFILE RUN SEQUENCES .....	99
27.3 METMANAGER RUN SEQUENCES.....	101
27.4 PARAMETER RUN SEQUENCES .....	102
27.5 COMPOUND (NESTED) SEQUENCES .....	107

<b>28 DISPLAYING OUTPUT .....</b>	<b>109</b>
28.1 CREATING CHARTS .....	109
28.1.1 <i>The Chart Specification dialog</i> .....	110
28.1.2 <i>Chart Properties</i> .....	113
28.1.3 <i>X-Axis Properties</i> .....	113
28.1.4 <i>Panel Properties</i> .....	115
28.1.5 <i>Y-Axis Properties</i> .....	115
28.1.6 <i>Series Properties</i> .....	117
28.1.7 <i>The Chart Window and its Menu</i> .....	118
28.1.8 <i>Printing Charts</i> .....	120
28.2 CREATING TABLES.....	120
28.3 CREATING MAPS.....	125
28.3.1 <i>Symbols Map Specification</i> .....	127
28.3.2 <i>Label Map Specification</i> .....	133
28.3.3 <i>Grid (Colour-Graded) Map Specification</i> .....	134
28.3.4 <i>The Map Window and its Menu</i> .....	137
28.3.5 <i>Printing Maps</i> .....	139
28.4 CREATING TEXT FILES.....	139
28.5 CREATING DATABASE TABLES .....	140
<b>29 THE MAP MANAGER FACILITY .....</b>	<b>141</b>
29.1 MAP REGION MANIPULATIONS .....	144
<b>30 THE METEOROLOGICAL DATA MANAGER FACILITY (METMANAGER) 146</b>	
30.1 <b>CREATE/EDIT/DELETE LOCATION SELECTIONS</b> .....	147
30.2 <b>CREATE A NEW DATA-FILE</b> .....	150
30.3 <b>OPEN AN EXISTING DATA-FILE</b> .....	150
30.4 <b>IMPORT METEOROLOGICAL DATA</b> .....	151
30.5 <b>EDIT LOCATION AND METEOROLOGICAL DATA</b> .....	153
30.6 <b>DELETE LOCATIONS</b> .....	154
30.7 <b>MAINTAIN THE DATABASE</b> .....	154
<b>APPENDIX 1. THE MODEL TRACE FILE.....</b>	<b>155</b>
<b>APPENDIX 2. THE LOC/MET FILE FORMAT .....</b>	<b>158</b>

# 1 Introduction

The dynamics of animal and plant populations are influenced by many factors, and understanding the response of a population to a multitude of external factors can be very difficult. Simulation models are a powerful means of representing such systems and allowing users to interact with them. These models help to summarize our understanding of a species' population dynamics, identify gaps in knowledge and enable rapid evaluation of management options. Building population models, however, can be expensive in time and require specialist programmer skills. The DYMEX package is designed to overcome the bottleneck caused by inadequate computer programming resources and modelling expertise. DYMEX enables the user to build a class of biological models referred to as mechanistic or process-based models, without the need to program a computer. It uses a modular modelling approach and consists of two parts: a **Builder** and a **Simulator**. The **Builder** is used to create and modify the model, while the **Simulator** is used to run a completed model and display the results of simulations.

This Simulator User Guide is concerned with the **Simulator**, while the **Builder** is dealt with in a separate Builder User Guide. The **Simulator** guide covers the following topics:

- Opening a model (page 4)
- Initializing the model (page 22)
- Adjusting parameters (page 76)
- Setting your preferences (page 83)
- Running the model (page 90)
- Displaying the output (page 109)

For users of both the Builder and Simulator, a series of 3 tutorials is available on the CD Rom: The *Insect*, *Annual* and *Perennial Plant Tutorials*. These allow the user to learn the basics of model construction in your field of interest by constructing a new model step-by-step. A comprehensive on-line Help system covering both the **Simulator** and **Builder** is also provided. It allows the user to look up the meanings of terms used in DYMEX, the functions of the various DYMEX modules and the available options and limitations. All the documentation is included on the CD-ROM and the manuals and tutorials can be read by installing *Acrobat Reader*, which is also on the CD-ROM.

Some conventions used throughout this guide are as follows:



The exclamation symbols throughout the manual represent *important notes*.



The light bulb represents *examples and ideas that can be used when setting up, running or displaying results from a model*.

## 1.1 What is the Simulator?

The DYMEX **Simulator** is a program that runs models created within the DYMEX **Builder**. Models created using the **Builder** are stored in files whose names have the extension .GMD (Generic Model Description files). To run a simulation using such a model, it is first opened in the **Simulator**. Initial conditions for a run can be set, and parameters altered as required. The simulation is then started, and at its completion, results can be tabulated, charted, or saved in a database table or text file. Sensitivity analyses can readily be performed on models using the automatic run sequences (see *Run Sequences*, page 94). For example, the effect of rainfall on the survival of a lifestage could be systematically changed by the program, and then some measure of population size (see *Run Summaries*, page 93) could be plotted against the changed rainfall parameters. The timing of events during the simulation can be systematically analysed by the program to give, for example, the optimal time to apply a pesticide. The population of the pest could also be plotted against the timing of pesticide application.

## 1.2 Differences between Versions 1 and 2

The Simulator has been extensively redesigned and there are numerous small differences from Version 1. The following list indicates the major differences:

- The paradigm used by the Simulator is altered considerably. In Version 1, the user opened a Model (GMD) file. In Version 2, the user opens or creates a new Simulation (DXS or INI) file. The Simulation File is essentially the same as the Initialization File used in Version 1 of DYMEX, and stores settings for a particular run. With this new paradigm, it is an easy matter to save simulation settings at any time.
- The Model Components window has been redesigned, with many new features. Many modules indicate their settings in more detail. Users can hide modules that do not require initialisation, thus simplifying the appearance of a model. More convenient access to parameters and information about the set of parameters is now available through this window. Sequences can be set more conveniently.
- The Parameter initialisation window has been enhanced, with a tree-view of the parameters within their hierarchy. A collection of parameter values for a particular module (Parameter Set) can be named, described and saved to a separate parameter file.



- New modules are available, as follows: Adjustable Circadian, Climate Change Scenario, Counter, Daydegree, Event with Delay, MetManager, Discrete QueryUser, Running Mean, Storage, Switch and Weather.
- The Lifecycle module has been considerably enhanced, allowing branching of lifecycles, nested stages (*Endostages*), an immigration process, and many more lifestage output types.
- Graphical output has been radically redesigned, with charts being more interactive, easier to set up and more flexible in the choice of options. Limits on number of panels and series in charts have been removed.
- The ability to produce maps has been added for multiple runs using sequences that produce results across geographic regions (for example, those controlled by MetManager or DataFile sequences).
- Table output is more flexible, allowing the table format to be modified “on-the-fly”. Tables from multiple runs can be sorted.
- Nested sequences, and enhanced capabilities for existing sequence types have been added.
- Results for individual runs within a multiple run (including detailed tables and charts) can be obtained from the multiple run output table.

### **1.3 Differences between Versions 2 and 3**

The following list includes the major differences. In addition, a number of minor improvements and bug fixes have been made to the Simulator program.

- Populations can be divided into separate sub-populations (demes) within the model to represent, for example, genetic types or spatial units. When this is done, variables and parameters that take part in the sub-population structure have components that correspond to each sub-population.
- The Lifestage window has been extensively redesigned to allow a much better overview of the processes and process components within each lifestage.
- Lifestages can be initialized with multiple cohorts. This is useful in cases where it is necessary to have each “cohort” correspond to one individual.
- Lifecycle now can contain factors that belong to the lifecycle as a whole (i.e., they are not part of lifestage processes). This can avoid redefining the same parameter multiple times.

## 1.4 Minimum Requirements to Run DYMEX

The minimum requirements to run DYMEX are:

- ◆ Pentium 400 MHz or better.
- ◆ Windows NT/2000 or Windows XP
- ◆ 256 megabytes of RAM.
- ◆ 200 megabytes of disk space.

## 1.5 Installing and Running DYMEX

### ➤ To install DYMEX on your machine

- 1.** Place the CD into the drive. The installation should start automatically. If it does not, continue with the following steps:
- 2.** Select **Run...** from the **Start** menu.
- 3.** Within the **Run** dialog box type **d:\setup.exe**. If the CD-ROM drive is not the **d:** drive, then substitute the appropriate drive letter for **d**.

### ➤ To run the DYMEX Simulator

- 1.** Go to the **Start** menu and find the **DYMEX** program group within the **Programs** group.
- 2.** Choose the Simulator icon to open the *Simulator* program.

## 2 An Overview of DYMEX Models

This section describes the fundamentals of a DYMEX model. It is important that the user understands these concepts to both understand the structure of a specific DYMEX model and interpret the results from a simulation. Many of the more general modelling issues are discussed elsewhere. See for example J.L. Goodenough and J.M. McKinion (eds.), *Basics of Insect Modelling*, ASAE monograph number 10, 1992.

A DYMEX model consists of a number of interconnected modules that describe the essential features of the system being modelled. Due to this modular structure, users familiar with DYMEX can quickly understand and use any model created in DYMEX.

### 2.1 Variables

State  
Variables

As with all models, a DYMEX model has a number of state variables that describe the state of the system at any particular time. Examples of such variables in a particular model may be *Maximum Temperature*, *Rainfall*, *Daylength*, *Total No. of Flies*, and *Egg Development Time*. Changes in the values of variables take place between one timestep and the next. The values of all DYMEX variables are updated by their associated modules. A history of values of the variables throughout a simulation run constitutes the results of

that simulation, and they are retained for tabulation or charting. All variables have names, and (with the exception of a few predefined names) the designer of the model is responsible for naming each variable.

One type of state variable that is commonly encountered in a DYMEX model is used to summarize or report on some internal model processes. For example, we might have a variable that outputs the average Physiological Age of individuals in a stage. This type of variable may not have defined states at all times during a simulation run (for example at times during the simulation when no individuals of that stage are present).

Delay  
Variables



Another variable type is used in DYMEX to describe intervals of time, for example the time required for an egg laid on a particular day to hatch. Such variables (termed *Delay Variables*) do not have their values assigned to them until the time period that they are monitoring has terminated (i.e., the eggs have hatched in the example). These variables are provided for use as output to graphs and tables. They include such commonly used lifestage outputs as Development Time and Cohort Duration.

Summary  
Variables

*Summary Variables* are used to summarize output variables into a single number. For example, a model might have an output variable that represents the population of a species of plant at weekly intervals (*No of Adult Plants*). A Summary Variable that gives the average population over (say) the last year of a simulation can be constructed from that output variable. This variable can then be used as a quick indicator of changes to the simulated population in response chemical treatments or parameter changes.

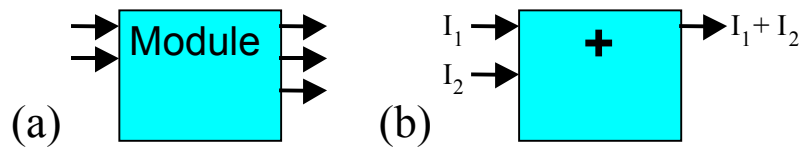
Demed  
Variables

In models that use sub-populations, some variables will contain multiple values for each time step, one for each model subpopulation (*Demed Variables*). The module that updates such a variable takes care of the details of appropriately updating each subpopulation component. When such variables are specified as variables for graphs or tables, the user will be prompted for the particular component that should be displayed. Note that *Summary Variables* that summarize demed variables will be similarly multi-valued.

## 2.2 Modules

It is important to understand the function of modules and how they interact in a DYMEX model. In its simplest form, a module is a black box that may have one or more input variables and has one or more output variables, as shown in Fig. 2-1(a). The module is responsible for calculating the value of its output variables at each timestep. Several different types of modules are available for use in a DYMEX model, each performing a different function. For example, an **Expression** module could have two inputs and one output, and calculate the sum of the two input variables, assigning the result to the output variable (Fig. 2-1(b)).

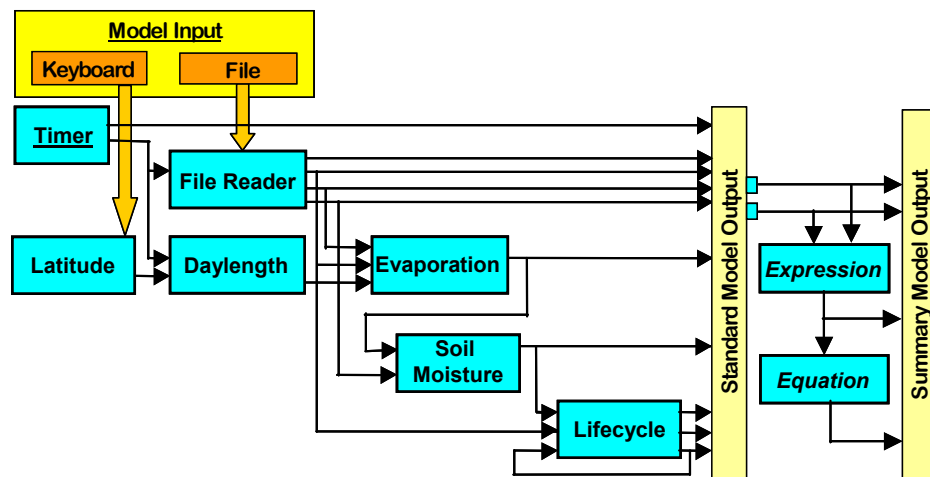
**Fig. 2-1 (a) Schematic representation of a DYMEX module; (b) a DYMEX “addition” module**



Modules are connected by having each of their input variables connected to an output variable of another (or the same) module. During a simulation, the value of that output variable is then used as the input value. Fig. 2-2 shows an example of a typical model and its links between modules. The model makes use of the **Soil Moisture** module, which has two input variables, Rainfall and Evaporation. The Rainfall input is linked to a File Reader (**DataFile**) module, while the Evaporation input comes directly from an **Evaporation** module. Note the feedback loop in the **Lifecycle** module, where an input variable is linked to an output of the same module.

It should be evident by now that when we refer to a module input, we are not referring to information read from a file or supplied via a dialog, but the points of connection to a “previous” module’s output variable. Data from a source external to the model (model input) such as data files or from the keyboard (through dialogs) is accessed via specialized modules (as shown in the Model Input block in Fig. 2-2). This information then becomes available via that module’s output variables.

**Fig. 2-2 A DYMEX model showing the flow of information (variables) between modules.**



All module output variables are available as part of the model output for tabulation, graphing, etc. Each of these outputs is an array of values that stores the variable’s values for each time step over the period of a simulation. Collectively, these outputs constitute the *Standard Model Output* (Fig. 2-2). Any of these output variables may be summarized in various ways. These summaries are available as *Summary Variables*, and the set of Summary Variables constitutes the *Summary Model Output* (Fig. 2-2). Some modules in

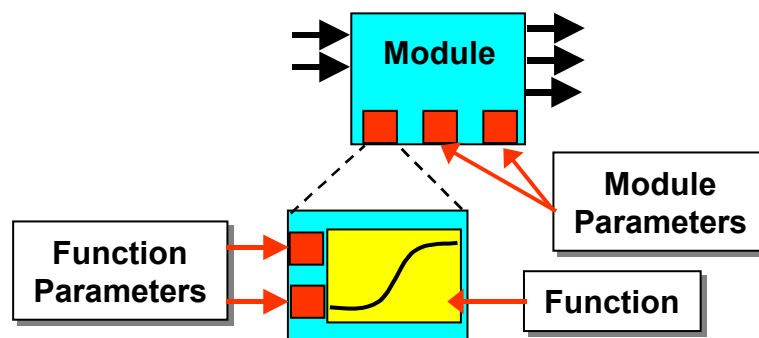
a model may manipulate these Summary Variables, creating more Summary Variables in the process (as shown by the two modules and their outputs at the right of Fig. 2-2). These **Summary Modules** are updated only once during a simulation run, directly after the simulation has finished running.

The model designer can configure most modules to adapt them to their required task. For example, Expression modules referred to earlier can be set up to add, multiply or average their input variables. The Lifecycle module in particular is extremely flexible, and can be set up in many ways to suit the particular organism being modelled. Several of the modules are sub-models, whose function is to do a specific job. For example, the 1-layer Soil Moisture module simulates the water balance in the top layer of soil. It is anticipated that one way that DYMEX will be enhanced in the future will be by the addition of new modules. The currently available modules and their application are summarized in Table 3-3.

## **2.3 Functions and Parameters**

Some modules can be adjusted by setting the values of one or more *parameters*. Parameters are somewhat similar to variables, except that their value is fixed during any simulation run. For example, the **Soil Moisture** module has 3 parameters, the Soil Moisture Capacity, the Evapotranspiration Coefficient and the Drainage Rate (Fig. 2-3 shows this schematically, with the parameters indicated as red squares). Thus parameters can be thought of as devices for adjusting a model to a particular place or situation.

**Fig. 2-3 A DYMEX module with 3 parameters, showing how a parameter can be replaced by a functional relationship**



The model designer may have replaced any DYMEX module parameter by a function or process. In the Soil Moisture example, we may have a relationship between the value of the Evapotranspiration Coefficient and some other variable (perhaps crop biomass). DYMEX allows us to substitute that relationship for the parameter (Fig. 2-3). In that situation, we end up with some additional parameters (the parameters of the functional relationship specified), as well as a new *de facto* input variable to the module (crop biomass). The term **Factor** is often used in DYMEX to refer to a component that could be either a parameter, function or process (see next section).

## 2.4 Processes

A **Process** in DYMEX is a mathematical relationship that results in a value at every timestep during the running of the model. Schematically, a Process can be described by the relationship

$$P(t) = f(a, b, \dots)$$

where,  $P(t)$  is the value produced at timestep  $t$  (often referred to as the **Process Rate**). The process rate depends on a series of components ( $a, b, \dots$ , in the equation) and these are usually referred to as **Process Factors** or **Process Components**. The  $f()$  relationship indicates that we combine the Process Factors in some way to obtain the Process Rate, and the exact form of the combination is the Process' **Combination Rule**. Much of the flexibility in DYMEX derives from the fact that **Parameters**, **Functions** and **Processes** are all Process Factors and can be used as the “ $a$ ”, “ $b$ ”, etc terms in the above equation.

Let us assume, for example, that we have a mortality rate (the Process Rate) that depends on temperature and moisture, and there is a constant mortality as well from (say) predation. Let us also assume that analysis of experimental data has shown that these mortality factors operate independently. This could be expressed in equation form as

$$m(t) = m(T) + m(M) + P$$

where,  $m(t)$  is the mortality during a particular timestep,  $t$ ,  $m(T)$  and  $m(M)$  are the mortalities due to temperature and moisture, respectively, and  $P$  is the mortality due to predation. Note the similarity to the earlier equation, with the Process Factors ( $a, b, \dots$ ) equivalent to the terms  $m(T)$ ,  $m(M)$  and  $P$ . In this case, the **Combination Rule** is a simple addition. To complete the specification of this Process in DYMEX, we could specify a **Function** with *temperature* as driving variable for the term  $m(T)$ , another **Function** driven by (say) *humidity* for the term  $m(M)$  and a simple **Parameter** for  $P$ .

Processes may be much more complex than the example given above, with up to 9 Process Factors allowed as components. Besides simple addition, multiplication, etc, a user-defined equation can be used as the Combination Rule, where required.

Processes are used in many types of modules, but are central to the operation of the **Lifecycle** module. Such *lifecycle processes* as development, mortality and stage transfer map directly to a corresponding DYMEX **Process**.

## 2.5 Sub-populations

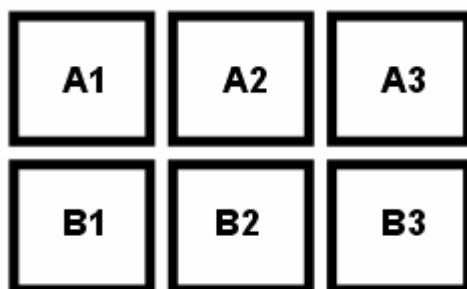
The individuals comprising a model population can be subdivided into subpopulations (these will also sometimes be referred to as “demes”). These subpopulations could represent different spatial units (for example, multiple fields with different crops, or riparian and upland locations) or they might

represent different genotypes. Both types of subpopulations can be simultaneously represented in a DYMEX model.

Internally, subpopulations are handled by creating variables that reflect the sub-population structure (*demed* variables). Consider, for example, a model that uses three sub-populations (say, A, B and C). Each variable that represents a property of the model population then consists of 3 components. For example, a variable named “Total Number of Larvae” would have components that gave the total number of larvae in sub-populations A, B and C. Parameters can also have a sub-population structure in such models. Much of the work in creating these sub-population variables is done automatically by DYMEX. All that the user needs to do is to specify the structure of the model when it is created in the **Builder**.

The sub-populations in a DYMEX model are structured into a number of dimensions. For example, the six sub-populations represented in Fig. 2-4 are organized into two dimensions (i.e., the sub-populations are structured as a 3x2 array. The horizontal dimension here may represent 3 genotypes, while the vertical dimension may, for example, be riparian and upland spatial units. Individuals in the sub-population labelled “A1” might be herbicide-susceptible weeds in the riparian site, while B3 would represent resistant weeds in the upland site. The sub-populations can be named and characterised as required for the particular model using the **Builder** program.

**Fig. 2-4 Sub-population structure showing six sub-populations, organized as two sub-population “dimensions”.**



Transfer of individuals between sub-populations is handled by a specialized type of process, the **Mixing Process**.

### **2.5.1 Genetic sub-populations**

In version 3 of DYMEX, if a genetic sub-population dimension is used in a model, it must be restricted to exactly three components. This corresponds to a single, autosomal locus with two alleles with Mendelian inheritance, so that each dimension component represents one genotype (AA, Aa and aa). Use of such a genetic dimension will automatically invoke a mixing process to simulate the mechanism of genetic exchange during reproduction (See the **Builder User's Guide** for details). The mixing process assumes random

mating. Genetic sub-populations could be used to address questions such as the best strategies to adopt to delay the development of resistance to a pesticide.

## **2.5.2 Spatial sub-populations**

Spatial sub-populations exist in different “patches” of the model domain. These might, for example, be adjacent fields, habitat patches or any spatial unit that the model domain may be divided into. Each spatial unit can be provided with coordinates, if required, that can be entered using a module such as “UserQuery”. If the properties of the spatial units vary, the model builder can provide for this by having environmental variables such as temperature or moisture that are different between the different units. The **Dispersal/Mixing Process** for spatial units (*Dispersal*) is used to distribute individuals between spatial units, with the timing of dispersal events specified by a **Dispersal Timing** process. A dispersal process can be placed into any lifecycle.



It must be mentioned that DYMEX is not designed to work as a GIS. The ability to model populations in separate spatial unit is intended to help answer questions about dispersal and movement on a local scale, with a limited number of such spatial units. This may include such phenomena as movement of moths between fields (perhaps with different cropping regimes or treatment schedules), seed dispersal down flood plains, movement of host animals between paddocks or even investigations on the effect of spatial heterogeneity on the stability of populations. The nature of the DYMEX *Lifecycle* module (specifically its cohort based mode of operation) will have the effect of rapidly increasing the computing resources required as the number of spatial units is increased. This will be compounded by the use of lifecycles with “endostages”.

## **2.6 DYMEX model files**

The Simulator makes use of several files to obtain model or parameter information and store operating settings. These are described below:

- **Model Description File (.gmd)**

The Model Description File (gmd file) contains a detailed specification of the structure of the model, consisting of a list of the modules, the internal structure and settings of these modules and the connections between them, as specified in the **Builder** when the model was built. The file also contains default values and the allowable ranges of the parameters. In some models, the model specification may be split between the main gmd file and one or more *Auxiliary Model Description Files* (gmi files). The gmd file is in a format that can be edited only with the use of the **Builder**.



- **Parameter File (.gmp)**

Each DYMEX model has a Parameter file, containing the values of each of the parameters in the model. The **Simulator** attempts to load a default parameter file (in the same location, and having the same name as the Model Description File, but extension *.gmp*) whenever a model is loaded. If a default parameter file is not found, the **Simulator** will create one using the default values from the *gmd* file. The current values of the parameters can be saved to a parameter file at any time, and a parameter file other than the default file can be used if required. Note that if the model is modified in the **Builder**, and then reloaded into the **Simulator**, the previously used parameter file may not work if parameters have been renamed, relocated or new parameters have been added. The **Simulator** will give a warning of this condition and allow a new parameter file with default values to be created. The user should not edit the Parameter File directly.

- **Simulation file (.dxs or .ini)**

The Simulation file (called the *Initialization File* in Version 1 of DYMEX) stores initialization settings for all modules. It also contains any user-defined chart, table, map and file formats, run summary definitions and run sequences. Simulation files are generally stored at the same location as the *gmd* file, but this is not a requirement. Each Simulation File is associated with a single model, but each model may have many Simulation Files associated with it. Simulation files may be saved at any time during a **Simulator** session. If a simulation file is not present, one must be created before a simulation can be run. When distributing a model, at least one Simulation file should be provided with the *gmd* file, to allow the user to run the model with the minimum effort. Users should not edit the Simulation file directly.


- **Simulator Configuration Settings**

Simulator Configuration settings and preferences that are applicable to any model are stored in the Windows® registry. These include all of the settings from the **Operating Preferences** dialog (Section 25.1).

### **3 Opening a DYMEX model**


Before a model simulation can be run, a simulation file must be loaded into the **Simulator**. Only one simulation file (and hence one model) can be loaded into the **Simulator** at any one time.

➤ **To create a new Simulation File**

- 1.** Either go to the **File** menu and select **New**, or if the button bar is displayed, left click on the **New** button .
- 2.** Select the model for which a new simulation is to be created from the list of models in the left panel. Note that you can verify that the selected model is the correct by using the data in the **Model Information** panel.

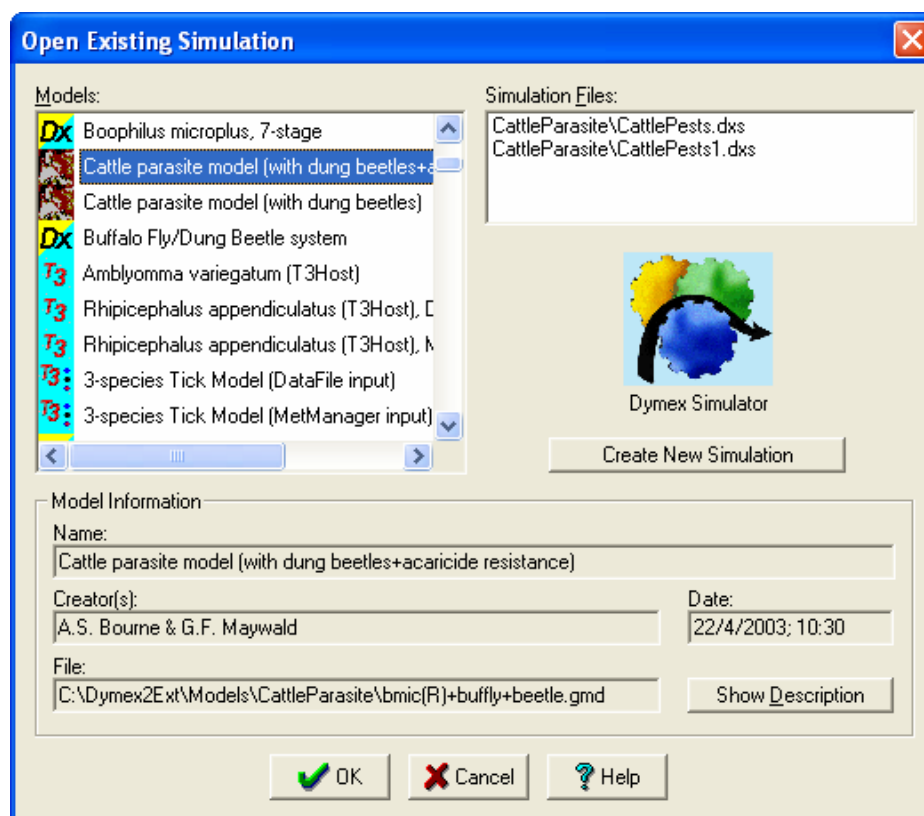
3. Type in a name for the new *Simulation File* into the box at the upper right. If the simulation file is to be created in a different directory, select the directory using the **Simulation File Location** selector (...).
4. Click Ok to create the new *Simulation File*.

➤ **To open an existing Simulation File**

1. Either go to the **File** menu and select **Open**, or if the button bar is displayed, left click on the **Open** button .
2. Select the model for which a simulation is to be opened from the list of models in the left panel (Fig. 3-1). Verify that that the selected model is the correct one by checking the data in the **Model Information** panel.
3. From the list of Simulation Files displayed in the **Simulation Files** panel at the right, select the required file by clicking on it. If only a single Simulation File is listed for the model, it can also be selected by double-clicking on the model name.

The **New Simulation** dialog can also be accessed from the **Open Simulation** dialog by clicking on the “**Create New Simulation**” button. The reverse is also true, in that the **New Simulation** dialog has an **Open Existing Simulation** button.

**Fig. 3-1 The Open Simulation dialog.**



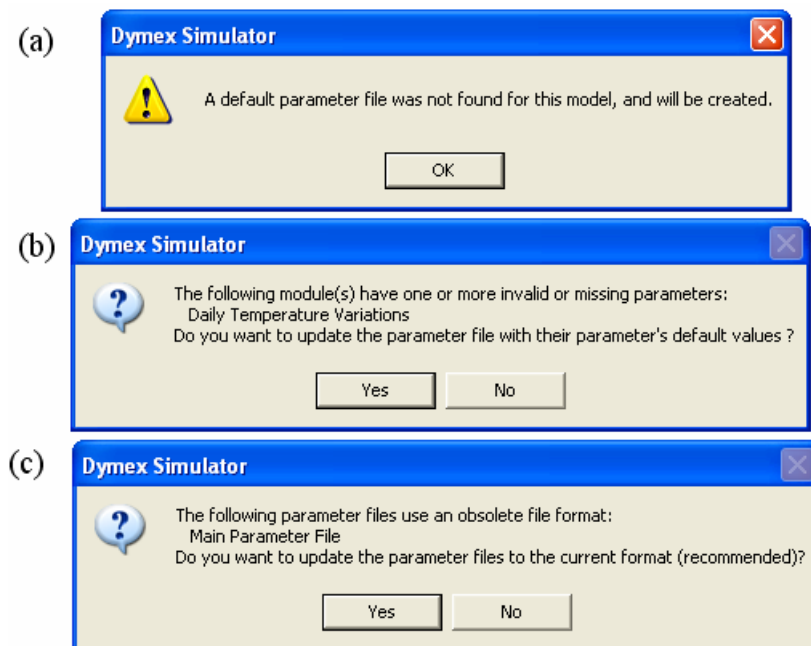
- ! Note that the models listed on the left side of both the **New Simulation** and **Open Simulation** dialogs are the gmd files found in the *Model Directory* (as set from the **Preferences** dialog). By default, the names of the models are listed. However, you can specify that the names of the model files be listed (go to the **Preferences** dialog to set this).

A small icon is shown just in front of the model name in the **Models** list. This is normally the default model icon (Dx). However, another icon can be associated with the model by placing a Windows bitmap file with the same name as the “gmd” file (but extension .bmp) into the same directory as the “gmd” file. The bitmap file can be any size – it will be automatically rescaled to fit into the allotted space.

- ! A recently used Simulation File can be quickly loaded by clicking on the **File** menu and then clicking on the Simulation File’s name (near the bottom of the file menu).

At this point, the selected Simulation File, along with its associated model will be loaded into DYMEX, along with its default parameter file, if one exists. If the parameter file could not be found (which will always be the case with a newly created model), a message box will appear (Fig. 3-2a) to inform the user of this. Clicking **Ok** will create a new parameter file containing the default values specified when the model was built.

**Fig. 3-2 DYMEX messages indicating that (a) no parameter file was found, (b) there are invalid or missing parameters in a module, or (c) the parameter file has an obsolete format.**



If the model has been edited in the **Builder** since it was last run in the **Simulator**, a message such as that shown in Fig. 3-2b may be encountered when it is next loaded into the **Simulator**. It indicates that the existing

Parameter file is inconsistent with the GMD file. Answering “Yes” will replace the parameters in the Parameter file for the indicated (“invalid”) modules with the default values taken from the GMD file. Answering “No” will leave the parameter file unchanged, but use the default values of the parameters from the GMD file. Parameters for the “invalid” modules cannot be adjusted in the *Simulator* in the latter case. The message shown in Fig. 3-2c will appear if the parameter file was created with an earlier version of DYMEX. The parameters will have been read correctly, but it is strongly recommended that they be saved in the current format (by clicking on **Yes**).

When the model is successfully loaded, a dialog box (the **Model Description** dialog) will appear containing the **Name** of the model you are using, the **Author**, the **Version** of the model, the **Date of Creation** of the model, the default **Timestep** used in the model and the **Comments** describing the model. Left click on the **Ok** to go on. The Model Description dialog may be bypassed by setting the appropriate option in the **Preferences** dialog.

At this point, the *Simulator* changes in appearance as a Component Window is created for the model. Some buttons and menu items will be disabled, while new menu options become available (see *Menu Commands*, page 17).

### **3.1 Errors during Model Loading**



The *Simulator* will check for errors while opening the model and a dialog box appears describing the problem if an error occurs (Fig. 3-3). Either one of two dialog boxes are used to report problems found during a model load operation, an **Error** or a **Warning** dialog box. The **Error** box appears when a serious error has been detected that prevents the successful loading of the model. A **Warning** box warns of anomalies or possible problems in the model or loading process (it could also indicate problems with the *Simulation* or *Parameter* files). In such a case, the **Continue** button can be clicked to continue opening the model. Note that these Error and Warning dialogs may also appear during or at the end of a simulation if any errors have occurred during the simulation. Within the dialogs, the following fields indicate details of the error.

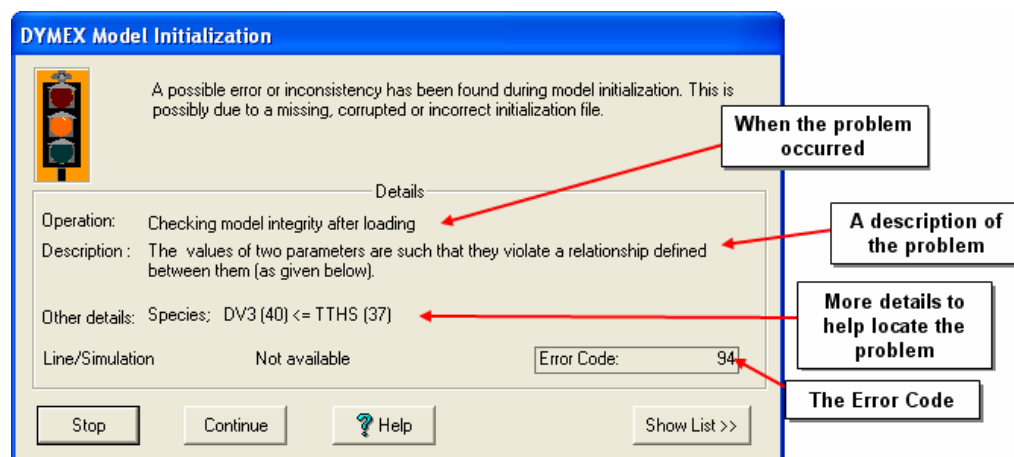
- Error Details
- Operation in Progress:** identifies the operation being carried out when the error occurred (for example, “Parsing Event module”). It gives a broad indication of where the error occurred during the load operation.
  - Description of Error:** gives a description of the exact problem that caused the error condition. For example, “An invalid type of 'Threshold Trigger' has been specified for an 'Event' module”.
  - Module or other Details:** lists the module or component where the error occurred if that information is available (for example, “Pesticide Application”).
  - Line Number:** reports the exact line in the GMD file where the error was found. Though users cannot edit GMD files directly, the information can be useful when reporting an error to Software Support. Some

operations cannot be localized to a particular line, in which case this field will contain 0 or a dash (-).

**Error Code:** should be reported to Software Support when reporting an error.

More detailed information on what some of the error messages mean and how the problem being reported could be fixed are available in the DYMEX Help System.

**Fig. 3-3** The error dialog box that is used within both the *Simulator* and *Builder*.



The Record  
Button

When a problem is encountered, the error dialog shows the first problem encountered at the highest error level (an **Error** is a higher level problem than a **Warning**). When a problem is encountered, the user may be able to obtain more information by clicking on the “**Show List**” button. This lists all the warnings and errors detected during the operation, in the order that they were found. The **Record** button, which allows the details given within the dialog box to be written to an error file (*dymex.err*), will also now be available. The information is appended to the error file, which will be located in the same directory as the GMD file. This is especially useful when calling for help about the error, or for later reference.

Very few errors should be encountered in practice, as the *Builder* will generally check for consistency of model structure, and not allow illegal operations. Errors will occur mainly if a GMD file is corrupted, or users have attempted to manually edit parameter or initialization files. Some common errors are explained in some detail in the Help system. Any unexplained errors may indicate defects in the software and should be reported to CSIRO Publishing Software Support.

## 3.2 The Model Components Window

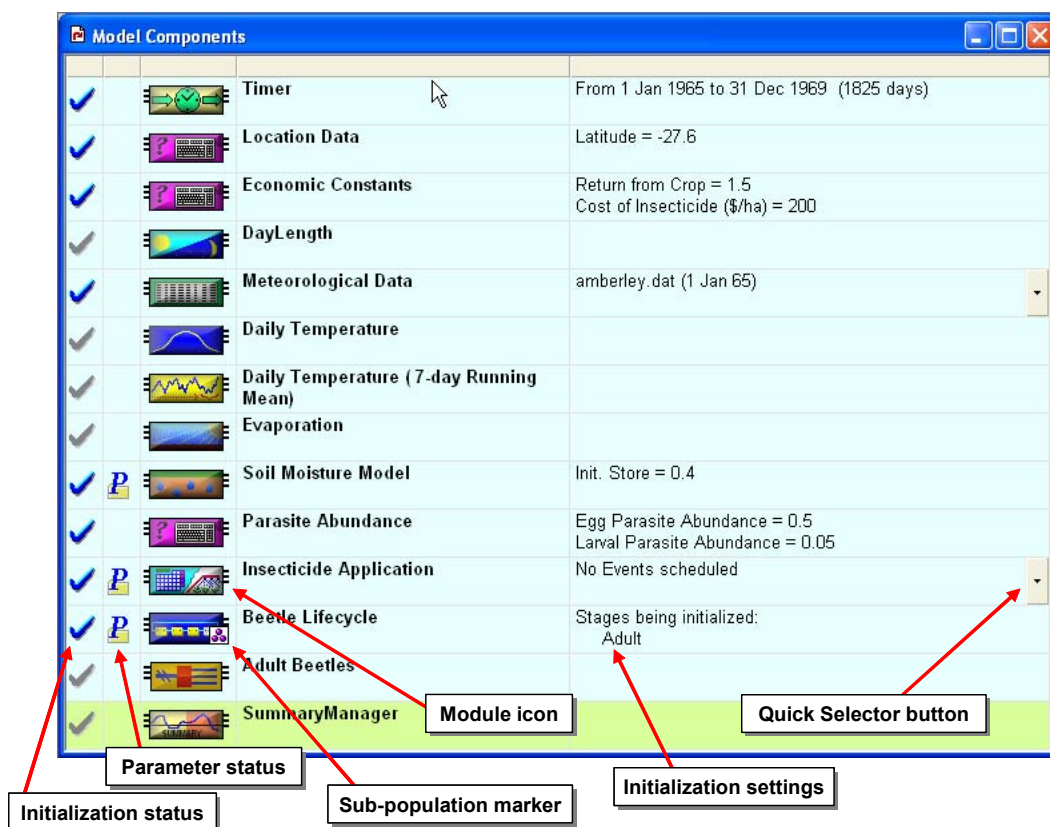
When a model is successfully loaded into the Simulator, it opens a **Model Components** window, which displays the model’s modules (Fig. 3-4). Many model operations are controlled from this window, where modules can be initialised, parameters modified and the model run. Generally, all the modules present in the model are shown in the window. However, the user may select

to shown only those models that have user-adjustable initialization settings. Another setting is available that will show the descriptions provided for the model and its modules in the Component Window. These settings are accessible from the *Model Preferences* dialog (Section 25.3). Several icons are used to display the type of module, its current initialisation status and the status of the associated parameters. Each module's current user-adjustable setting is displayed in the right-most column of the window. Some modules display a "Quick Selector" button at the far right, which can be used to quickly select Sequences (see Section 27) or parameter sets (see Section 24).

When the modules are all correctly initialised, a simulation can be run. Each completed simulation run creates a Run Window, from which the results can be examined, either as tables or charts (or maps for some types of multiple simulations).

Note the presence of the **SummaryManager** module at the bottom of the module list in Fig. 3-4. This module is automatically created for all models and manages any Summary Variables specified for the model.

**Fig. 3-4 The Component Window, showing a loaded model.**



All *Simulator* operations are available from the menus. Alternatively, many operations can be performed by placing the mouse cursor over the relevant module icon and left clicking the mouse. Clicking on a Parameter status icon opens the Parameter setting dialog, showing the parameters for that module.

Note that the menus (and the availability of buttons) change in accordance with which window is currently active. Most activities can only be performed if the corresponding window is active. For example, a graph can only be created when a Run Window is open following execution of a simulation when the results are available.

### 3.3 Menu Commands

Many of the *Simulator* commands can be performed from the menu bar. Below in Table 3-1 is a brief definition of all the menu commands.












**Table 3-1 A brief description of the function of the menu commands.**

Menu	Description of function
File	<b>File</b> enables Simulation Files to be opened & closed, and gives access to the MapManager, printing and allows the program to be closed.
View	<b>View</b> enables the model description, subpopulation structure and lifestage processes to be viewed.
Initialization	<b>Initialization</b> enables the initialization of the modules of the model and modification of the model's parameters (see <i>Initializing Modules</i> , page 22).
Execution	<b>Execution</b> enables the model to be run, and cohort removal conditions, run summary settings and run sequences to be defined (see <i>Running the Model</i> , page 90).
Preferences	<b>Preferences</b> enables the operating preferences, such as appearance of the window, and model tracing options to be modified (see <i>Simulator Preferences</i> , page 83).
Results	<b>Results</b> is activated when a simulation has been completed and offers options which allow charts, tables etc. to be created (see <i>Displaying Output</i> , page 109).
Chart	<b>Chart</b> is activated when a chart is displayed and gives access to chart operations (see <i>Creating Charts</i> , page 109).
Map	<b>Map</b> is activated when a map is displayed and gives access to chart operations (see <i>Creating Maps</i> , page 125).
Window	<b>Window</b> allows switching between and arrangement of open windows within DYMEX.
Help	<b>Help</b> opens the Help system and provides information on the version of DYMEX being used.

The following buttons are available from the toolbar. If a function is unavailable at a particular time, that button will be grayed (disabled).



**Table 3-2 A description of the main functions of the buttons of the tool bar.**

<i>Tool Bar Item</i>	<i>Description of function</i>
	Create a new Simulation File
	Open an existing Simulation File (see page 4).
	Save Simulation File or Parameter File (see Modifying Parameters, page 76).
	Run simulation (see Running the Model, page 90).
	Create chart (see Creating Charts, page 109).
	Create table (see Creating Tables, page 120).
	Create map
	Create a file of results
	Print Parameters/Chart/Table.
	Print Preview
	Help.

### **3.4 Model Description**

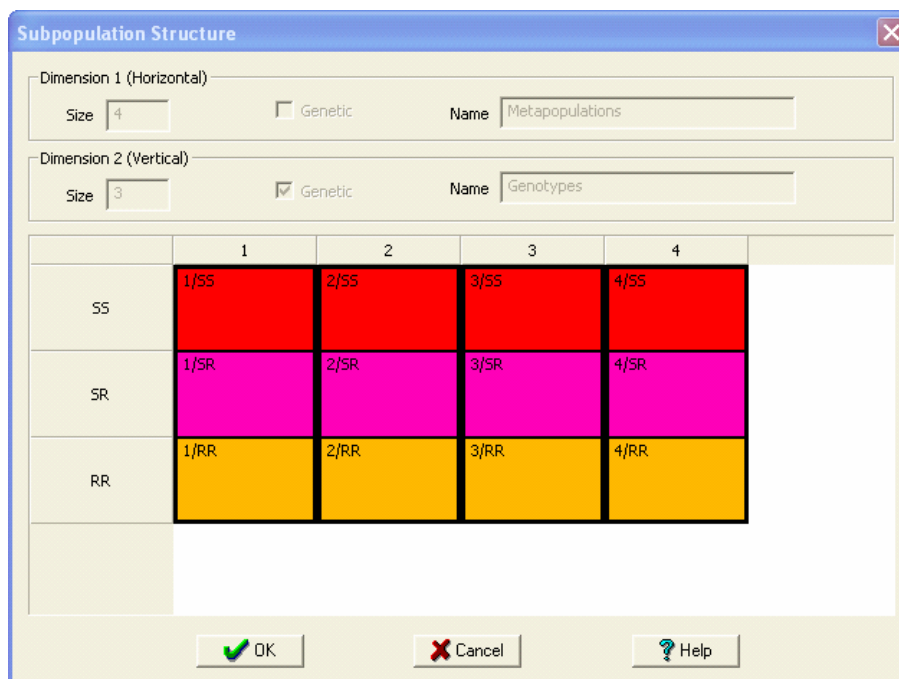
The **Model Description** dialog is accessed by selecting the **Model Description** option in the **View** menu. Information available in this dialog includes the name of the model, its author as well as version details such as the version number and date when the model was last modified. The major portion of the dialog is taken up by a general description of the model (provided by the model designer). Assumptions made in constructing the model will often be listed here. The **Model Description** dialog can be displayed automatically every time a model is loaded by setting the appropriate option in the **User Preferences** (see *Operating Preferences*, page 83).

### **3.5 Model sub-population structure**

If the model population is divided into subpopulations, this sub-population structure may be viewed by selecting the **Subpopulation Structure** option from the **View** menu. The menu option will be disabled if the model is a single-population one. The resulting dialog displays each subpopulation as a coloured rectangle (Fig. 3-5).



**Fig. 3-5 The Subpopulation Structure dialog box showing 4 spatial subpopulations, each further divided into 3 genotypes.**



If genetic subpopulations are used, 3 different colours indicate the different genotypes. If only spatial subpopulations used, a single colour (green) is used for all the subpopulations. A similar dialog is used in the Builder, where the subpopulation structure can be changed by the user. Note that the names of the subpopulation “slices” are shown as column and row headings. The names of each subpopulation is a combination of the “slice” names.

### 3.6 Printing the model

The model in the Component Window can be printed from the **File** menu by selecting **Print**, after making sure the Component Window is the active window. Check that the option to display descriptions has been set in the **Model Preferences** dialog (see Section 25.3), so that the printed document will include the description of the model and modules provided by the designer. Printing of other model details is available from other windows, in particular the parameters and their values from the Parameter Window (see *Modifying Parameters*, page 76), results of simulations from the Run Window, and tables, charts and maps from their windows.

Note that a **Print Preview** feature is available for most printing options. This is especially useful for windows such as the Component Window, which may be too wide to fit across a page. Many printers provide a scaling feature (available from the **Print Setup** menu item), which can be used in conjunction with the **Print Preview** to scale the “table” so that it fits across the page. Alternatively, the Components Window could be printed in *Landscape* mode.




Note that a much more detailed model printing facility is available from the DYMEX Builder program.

**Table 3-3 Module names and icons.**

<i>Module Icon</i>	<i>Module Type</i>	<i>Subpopulations</i>
	Timer	
	DataFile & MetBase (Meteorological Database)	
	MetManager (Climate Database)	
	Circadian & CircadianAdjust	
	Lifecycle	✓
	QueryUser & QueryFile	✓
	Event and EventWithDelay	✓
	Expression	✓
	Function	✓
	Equation & Counter	✓
	Running Mean	
	Storage	
	DegreeDay	
	Soil Moisture (1-layer)	✓
	Evaporation	
	Daylength	
	Climate Change Scenario	
	Switch	✓
	SummaryManager	
	DemeSplitter, DemeStatistics	

## 3.7 Modules

There are 19 different types of modules that can be used in a DYMEX model. Each type of module has a different function. The bitmap that represents each module has an appearance indicating the module's function. For example, a **Lifecycle** module has a drawing of a lifecycle within it (Table 3-3). Note that in models with multiple subpopulations, those modules that have output variables with separate components for the subpopulations are indicated with a *sub-population marker*,  (Fig. 3-4).

The creator of the model gives each instance of a module a unique name, which should indicate its purpose. Table 3-3 lists all the types of modules available in DYMEX, along with their pictorial representation.

In the **Model Components** window (Fig. 3-4), each module is displayed on one row of a table. The bitmap representing the module appears in the third column of the table, followed by the module name. The type of module represented is shown in a small popup window when the mouse is placed over the module bitmap. The rightmost cells in each row contain a summary of the current initialization setting for those modules that have settings that can be changed in the Simulator.

### 3.7.1 Module Information

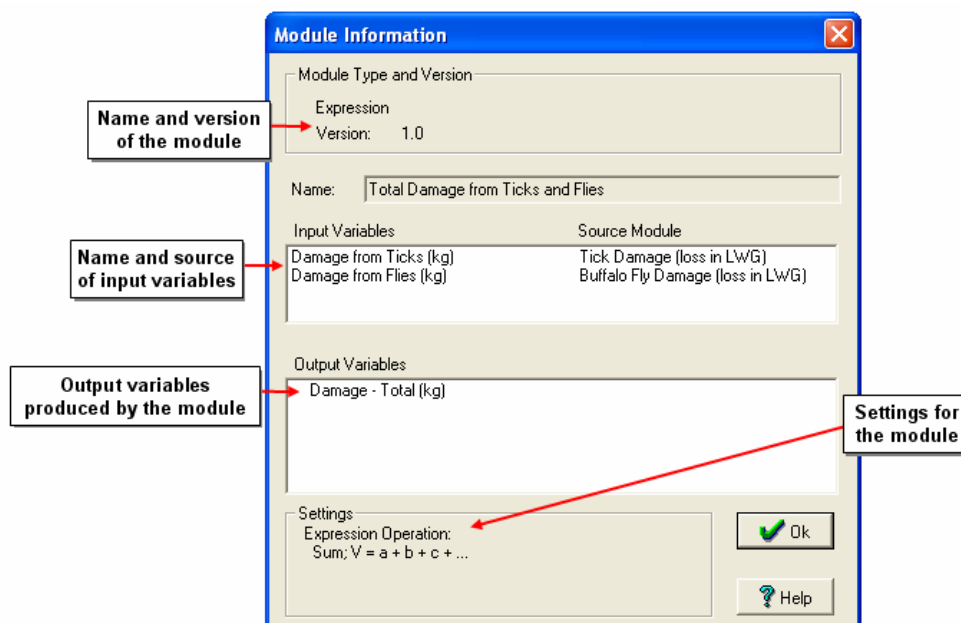


**Module information** is a facility that is available for all modules when clicking on the corresponding bitmap. This supplies information on the name and version of the module. It also lists the module's **Input Variables** and **Output Variables**. The listed input variables include not only the module's direct input variables, but also those variables that are inputs by virtue of the fact that they are driving variables of module functions. The list of input variables includes the source module for each variable (i.e., the module which provides the value for that variable). Finally, the **Settings** of the module are listed in the dialog. These are module properties that have been chosen by the designer of the model and cannot be changed by the user without recourse to the **Builder**. Note that if the model designer has provided descriptions for the module's output variables, these can be viewed by clicking on the desired output variable name in the **Output Variables** list-box.

The module information from an Expression module is shown in Fig. 3-6, listing its input variables and their source modules and the operation performed on those variables within the **Settings** box and the name of the output variable.

The following sections describe the functions of each module, along with details on how they are initialized. More detail about each of these modules can be found in the **Builder** User's Guide.

Fig. 3-6 The Module Information dialog box for an Expression Module with example module information.



### 3.7.2 Initializing Modules

Before a simulation run is started, the model must be initialized. Initialization consists of setting various values and options to those required for a particular run of the model. Some initializations are required before a simulation is possible. Most initializations relate to a particular module, and the state of initialization is indicated by a “tick” (✓) mark in front of the module bitmap.

- ✓ • A **grey tick** indicates that the module cannot be initialized – i.e., there are no user-adjustable settings (with the possible exception of parameters).
- ✓ • A **blue tick** indicates that the module has user-adjustable settings and these are currently set to values that allow a simulation to proceed. Note that a blue tick does not mean that the user-adjustable settings are correct, or will yield the required results. For example, a lifestage is initialized even with no individuals added to the population of any stage, in which case the population will remain zero throughout the simulation.
- **The absence of a tick** indicates that the module has user-adjustable settings, which are currently either not set, or are set to values that are incompatible with a simulation proceeding. The initialization settings need to be set to valid values, when a blue tick will result.

To Initialize a Module

To initialize a module, centre the mouse over the module icon and left click. This will display a list of options, which will vary with the chosen module. The **Module Information** option will always be present. If the module has associated parameters, there will also be a **Show Parameters** option as well as a parameter icon in front of the module bitmap. The Initialize Module option will be present only for those modules that have initialization options that can

be changed. Choose **Initialize Module** to initialize the module. Initialization settings will be stored in the model's *Simulation File*, and therefore do not need to be set again for subsequent simulation runs (even if the user leaves DYMEX between runs).

## 4 The Timer module



The **Timer** module supplies the basic timestep used by modules within the model. The duration of the simulation has to be set in this module before a simulation can be run. The **Timer** performs the timekeeping functions of the model. When a new model is created in DYMEX, the Timer module is automatically added to the model. The Timer is a mandatory component of the model and is the only type of module that is present in every DYMEX model.

A single, optional, input variable may be used by a **Timer** module: the “**Equilibrium Variable**”. If used, equilibrium runs (i.e., simulations that run until the model reaches a steady state) can be specified for the model. The conditions that define equilibrium need to be set (see *Running to Equilibrium*, page 90).

### Output Variables

Any of four output variables may be available from the **Timer** module: **Days Since Start**, **Day of Year**, **Simulation Date** and **Time of Day**. **Days Since Start** is the number of days since the start of the simulation. **Day of Year** is the number of days that have passed since the start of the current year (i.e., from, and including, January 1 to the current simulation date) in the simulation, and **Simulation Date** is the date currently being simulated. **Time of Day** is available only for model with a 1-day timestep using segmentation. It gives the number of hours that have passed since the start of the current day in the simulation.

### 4.1 Timer Initialization

Initialization of the **Timer** module consists of setting the length of time for which the simulation will run in the **Simulation Duration** dialog (Fig. 4-1). If a Simulation Date is used within the model, then the exact starting date for the simulation run needs to be set. If one or more data files are being used, the beginning date and end dates of the simulation must lie within the time of overlap of all data files, otherwise any date and time length can be used.

#### ➤ To initialize the Timer module

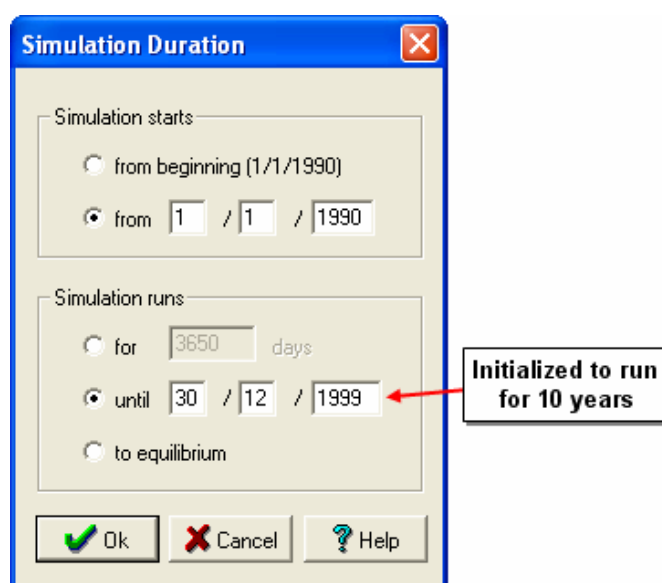
1. Left click on the module and select **Initialize Module**.

The appearance of the **Simulation starts** area of the dialog will vary with the type of model that has been loaded and whether other initializations have already been completed according to the following:

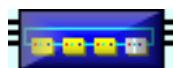
Day of Year or Simulation Date variable used	DataFile/MetBase modules already initialised	from beginning	from specific date
no	-	available	not available
yes	no	not available	available
yes	yes	available	available

2. From the **Simulation starts** box choose whether you want the simulation to start from the **beginning** or from a specific date (Fig. 4-1). If a specific date is chosen, fill in the date in the boxes provided (the format is day/month/year).
3. From the **Simulation runs** box choose how many days the model is to run for or the last day of the required simulation period (Fig. 4-1). Normally, the **to equilibrium** option is unavailable. However, users can specify the special conditions needed to allow a model to run to equilibrium (see *Running to Equilibrium*, page 90), in which case the **to equilibrium** option becomes available and can be selected as an alternative to specifying the run duration in days.

Fig. 4-1 The Timer initialization dialog box.



## 5 The Lifecycle module



The **Lifecycle** module defines the structure of a lifecycle in terms of *lifestages*, their interconnections and the *processes* within them. The structure of the lifecycle can be viewed by left clicking on the lifecycle module bitmap and selecting **Show Lifecycle Diagram**. The user specifies where and when individuals are introduced into the simulation to initialize this module (see *Lifecycle Initialization*, page 34).

What is a Lifestage?

A **Lifestage** contains all of the individuals in a simulation that are in a particular developmental stage, whether development is measured in terms of chronological age, physiological age, size or some other measure. DYMEX is

particularly suited to modelling insects, which do not have continuous development. The cuticle, which makes up the exoskeleton of insects, prevents continuous growth. A moult has to occur at intervals to form a new cuticle of larger surface area, so intervals between moults form a natural division of the lifecycle into stages.

Lifestages in DYMEX can also be used to model organisms with continuous development patterns. Multiple lifestages can be defined in many other organisms, e.g. frogs (eggs, tadpoles, adults), plants (seeds, seedlings, juveniles, small reproductives and large reproductives) and so forth. For less detailed models, a DYMEX lifestage could include several consecutive developmental stages.

In models that use more than one subpopulation, all or part of the Lifecycle can take part in this subpopulation structure. Individual lifestages can be specified to use only a single subpopulation. In that case, when individuals transfer from a lifestage that uses multiple subpopulations to one that does not, all the individuals from the different subpopulations are combined into a single subpopulation.

## 5.1 Cohorts and Cohort Properties

DYMEX does not model the fate of individual organisms. Individuals are grouped into assemblages termed *Cohorts*, where each cohort consists of a number of individuals that belong to the same lifestage, occupy the same spatial unit, and share the same properties, like the time (day/week) they entered a stage. Cohorts are the basic units that are modelled in a DYMEX lifecycle. An example of a cohort would be all the juveniles born on a particular day in a particular spatial unit during the simulation. All the individuals within a cohort experience the same conditions during the course of a simulation. .

**Table 5-1 Cohort Properties and their associated processes**

<i>Cohort Property</i>	<i>Associated Processes</i>
<b>Number</b>	Mortality Stage Transfer Dispersal
<b>Chronological Age</b>	<i>Intrinsic</i>
<b>Physiological Age</b>	Development, Aging, Maturation
<b>Residual Fecundity</b>	Fecundity Progeny Production
<b>Size</b>	Growth

Cohorts have a number of properties (*Cohort Properties*) that are shared by all the individuals in the cohort. A list of these properties is provided in Table 5-1. For example, **Number** contains the current number of individuals in the cohort, while **Physiological Age** contains their current state of development. The values of Cohort Properties are changed during a simulation run by the associated processes. For example, the result of the Development process is to change the value of the **Physiological Age** Cohort Property. Some Cohort Properties are affected by more than one process. There is no user-adjustable process associated with the **Chronological Age** property, which always reflects the either the number of timesteps or the number of days since the cohort was created. The **Residual Fecundity** Cohort Property is only present in reproductive stages. Note that **Size** is not a pre-defined Cohort Property, but has been defined by the model designer. Simple lifecycles may contain no user-defined Cohort Properties, while complex lifecycles may contain many. The model designer also defines the processes associated with user-defined Cohort Variables. See *Examining the Cohort Properties* (page 41) for how to obtain a list of all the Cohort Properties in a lifecycle

A Cohort Property can be *global* or *local*, with the former keeping its value when individuals transfer between lifestages, while the latter is reset to its initial value when a new cohort is created. Global cohort properties need to have their values reset at some point during the lifecycle. Note that when individuals transfer between different spatial units in a simulation, the values of both local and global cohort properties are retained.

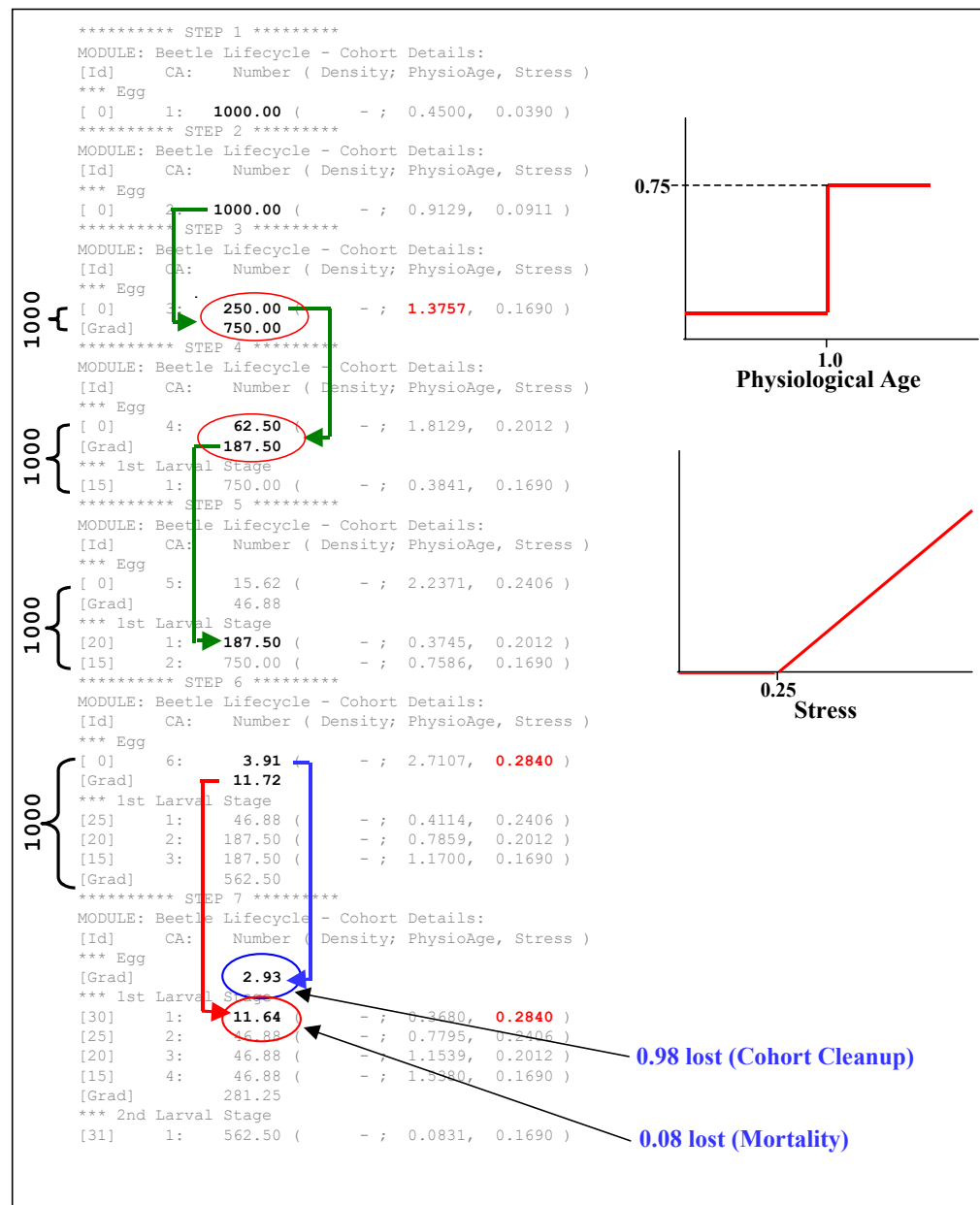
At any one time during a simulation, each lifestage may contain many cohorts. The total number of individuals in a lifestage is the total of the **Number** Cohort Property for all the cohorts in that stage.

How do  
cohorts work  
in practice?

Say we have a model containing a multi-stage insect lifecycle, the first two stages of which are “Juvenile” and “Adult”. We might initialise a simulation by introducing 100 juveniles into the simulation on day 1. These 100 juveniles would form a cohort. During the course of the simulation, the **Development** process chosen by the user acts on this cohort, increasing its **Physiological Age**. At any timestep, each member of the cohort has the same **Physiological Age**. If a **Mortality** process has been chosen to apply to the “Juvenile” stage, this mortality process acts on the cohort to reduce its numbers from the initial 100. If mortality is such that the number of individuals in the cohort is reduced to 0 before they are ready to move to the next stage, the cohort is automatically removed from the simulation. If the mortality is lower, the cohort may persist long enough for its remaining members to become eligible to move on to the next lifestage (as determined by the appropriate **Stage Transfer Process**). In this case, a proportion of individuals from this cohort (determined by the **Stage Transfer Process**) graduate to the next stage (“Adult”), where they contribute to creating a new cohort. This new cohort will consist of all the new adults for that timestep, recruited from a number of different juvenile cohorts, with early transfers from later juvenile cohorts added to late transfers from earlier juvenile cohorts.



**Figure 5-1 Extract from DYMEX “Model Trace” output showing cohort dynamics, and movement of individuals between stages.**



We can use the example model provided with DYMEX (“beetle.gmd”) to examine in some detail how cohorts actually work. The model simulates a beetle population. The beetle lifecycle consists of an “Egg” stage, several larval stages, and a pre-adult and adult stage. To examine the action of cohorts, we will focus on the first 2 stages, “Egg” and “1<sup>st</sup> Larval Stage”. Development of eggs is a function of temperature, with “Physiological Age” corresponding to the state of development as usual. The transition (hatching) function is a simple “Step” function, with 0.75 of eggs with the required Physiological Age of 1 hatching each timestep. Egg mortality is handled by accumulating “Stress” during the egg stage, and applying this as an establishment mortality to reduce the number of larvae hatching. This mortality applies after eggs reach a stress level of 0.25. After loading the model into DYMEX, we initialize it with 1000 eggs at the start of simulation,

and run it with the Amberley meteorological data (starting on 1/1/1965), and with “Model Trace” enabled for the lifecycle (Select the “Cohort Contents” and “Show only Lifecycle modules” options in the **Model Trace Options** dialog). This run produces the detailed results shown in Figure 5-1.

In the section labelled “STEP 1”, we have the results of the simulation after one timestep (ie, *at the end of timestep 1*). Each cohort in the simulation is shown on one line starting with an identifier (in square brackets, i.e. [0]) and the current values of the Cohort Properties. Note that the actual value of the cohort identifier has no significance. Thus, at the end of the first timestep, our single cohort, labelled [0], has a Chronological Age (CA) of 1, contains 1000 individuals (Number), and has a Physiological Age (PhysioAge) of 0.45 and Stress of 0.039.

*At the end of timestep 2*, we still have the same cohort, with the values of Physiological Age and Stress increased to 0.9129 and 0.0911, respectively. Note that the Physiological Age is still below the value of 1 needed for hatching.

*At the end of timestep 3*, we have a changed situation. During that timestep, the Physiological Age of the cohort increased to above 1 (it is 1.3757), and therefore the hatching function was invoked, which created a “graduate” cohort of 750 individuals (0.75 of 1000). These graduates will be used to create a new cohort of “1<sup>st</sup> instar Larvae” (cohort [15]) at the beginning of the next timestep. The remaining 250 eggs stay in Cohort [0] and accumulate more Physiological Age and Stress.

*At the end of timestep 4*, another 75% (187.5) of the remaining eggs in cohort [0] move to a graduate cohort [20]. The graduate cohort from the previous timestep is now a larval cohort [15] (and has accumulated a Physiological Age of 0.3841 during this step). A total of 62.5 eggs remain unhatched in the original cohort.

*At the end of timestep 5*, another 75% (46.88) of the remaining eggs in cohort [0] move to a graduate cohort [25], and only 15.62 eggs remain. We also now have two larval cohorts, each with their own values of Physiological Age and Stress. Note that until this point no mortality has occurred (and no individuals have been added through reproduction), so that the total number of individuals in all cohorts add up to 1000.

*At the end of timestep 6*, the egg cohort [0] still contains 3.91 individuals, while 11.72 graduates have been moved to a new graduate cohort. Note the Stress value that has been accumulated by cohort [0] now – it is 0.2840, which exceeds the threshold stress of 0.25. This stress is transferred to the 11.72 graduates from that cohort, and results in only 11.64 individuals in the larval stage cohort [30] formed from these graduates at the beginning of the next timestep, the rest (0.08) are lost through the establishment mortality process. It is worth noting here that Stress is a global Cohort Property and so its final value in the egg stage (0.284) is transferred to the new larval stage cohort. During timestep 6, the oldest larval cohort (labelled [15]) has reached a

Physiological Age of 1 and formed a graduate cohort. These graduates will form a new cohort of 2<sup>nd</sup> stage Larvae at the beginning of the next timestep.

*At the end of timestep 7, the egg cohort [0] has gone. 75% of its previous 3.91 individuals have formed the new graduate cohort. The other 0.98 individuals are lost to the system through the process of “Cohort Cleanup”, where cohorts containing a small residual number of individuals are removed from the simulation.*

This example illustrates how the DYMEX “bookkeeping” system works to track individuals through cohorts from one stage to the next. Individuals can only be lost to the system through an explicit mortality process (or via the “Cohort Cleanup” mechanism). Individuals are added to the system either through stage initialization (as with the 1000 eggs above), or through a reproductive process.

The cohort concept is central to the operation of the DYMEX Lifecycle module. In practice we have found that many problems in interpreting DYMEX results are due to a lack of understanding of how cohorts work.

## **5.2 Lifestage processes and their characteristics**

As described in Section 5.1 above, lifestage processes are the mechanisms by which *Cohort Properties* are changed during a simulation. They represent the dynamic part of the lifecycle module. For example, the process of **Development** increases the value of the **Physiological Age** at each timestep, while a **Mortality** process removes individuals from the cohort (i.e. it decreases the value of the cohort property called **Number**). Some cohort properties may have more than one process acting on them. A typical example is the **Number** of individuals in the cohort. **Mortality**, **Stage Transfer** and **Dispersal** processes all affect this property.



*Since processes are so fundamental to the operation of a DYMEX model, a considerable amount of flexibility has been built into them. It is important to understand the structure of the process in order to build a useful model.*

Each lifestage process is implemented as a normal DYMEX Process (see Section 2.4). It consists of one or more **Factors** (or **Process Components**), which are evaluated separately and are then combined using a **Combination Rule** to arrive at a final value (the process rate,  $r$ ). Process Factors can be either static **Model Parameters** (which do not change their value during a simulation run), or dynamic factors (**Functions** or **Processes**). Factors that are functions will have a specified shape and a number of Model Parameters. Factors that are processes will have their own component factors (which in turn can be either Model Parameters or Functions). The composition of lifestage processes can be examined either from the Parameter window (see *Modifying Parameters*, page 76) or the Lifestage window (see *The Lifecycle Diagram*, page 38).

Note that the output (value) of a process is a *rate* (per timestep).

The process rate is used to change the value of its associated Cohort Property in one of four ways (termed the **Update Method**):

- **Direct** update is an incremental update method where the value of the associated process ( $P$ ) in the any timestep is added to (or subtracted from, if updating is “inverted”) the value of the Cohort Variable from the last timestep ( $V_{t-1}$ ) as in the equation below. An example where the **direct** update method would be used is *Stress*, where *Stress* is accumulated throughout the lifestage. *Physiological Age* also uses the direct update method.

$$V_t = V_{t-1} \pm P$$

- **Proportional** update considers the value obtained from the process components in the current timestep ( $P_t$ ) to be a proportion. This proportion is multiplied by the cohort variable value in the last timestep ( $V_{t-1}$ ), with  $V_{t-1}$  incremented (or decremented, if the “inverted” option is set) by the result to obtain the new value of the cohort variable ( $V_t$ ), as in the equation below. Mortality is an example of an **Inverse proportional** update, where the values obtained from the process components are multiplied by the value of Number in the previous timestep and that value is removed from Number. For example, given a population in the previous timestep of 100, if there is a mortality of 0.8 then there will be only 20 left at the end of the current timestep.

$$V_t = V_{t-1} \pm (V_{t-1} \times P_t)$$

- **Current Value** update uses the value obtained from the process ( $P_t$ ) in the current timestep as the new value of the Cohort Variable ( $V_t$ ).

$$V_t = P_t$$

- **Current Average** updates the Cohort Variable ( $V_t$ ) with the running average of the process values over the life of the cohort. **Current average** cohort variables are always local variables, initialised to 0 ( $V_0$ ) when a cohort is created. Thus, if  $P_t$  is the process value for the current time step, and  $t$  is the age of the cohort (in timesteps), then:

$$V_t = \frac{P_t + V_{t-1}}{t}, \quad V_0 = 0$$

## 5.2.1 Mortality



**Mortality** is a process in the lifecycle module of DYMEX that acts on the **Number** cohort property. The mortality value is the **proportion** of the cohort dying in one timestep, whether it is one day, one week or one month. Three types of mortality process may be present in each stage – **establishment**, **continuous** or **exit**. Establishment mortality acts only at the time when the cohort is created, while continuous mortality is applied at every timestep

during the life of the cohort. Exit mortality acts only at the time individuals leave a cohort. In models that use spatial subpopulations, an additional mortality process (**dispersal mortality**) is available. It acts only on those individuals that have undergone dispersal during a particular time step, just after the dispersal is completed. In all other respects, it is the same as the other mortality processes. The **Complement Product** combination rule ( $R = 1 - (1-a) \times (1-b) \times \dots$ ) is generally used to combine mortality process factors.

## 5.2.2 Development



**Development** is the process that updates **Physiological Age** during each timestep. **Physiological Age** is not transferred between lifestages (i.e., it is a *local Cohort Property*) and starts from 0 within each new cohort. Generally, **Physiological Age** will be scaled so that a value of 1 indicated completion of development, though in degree-day type models, it may be scaled so that a value equal to the required degree-days for full development represents completion of development.

## 5.2.3 Dispersal Timing

The **Dispersal Timing** process is available only in those lifecycles that use spatial subpopulations. Its value at any time specifies the proportion of individuals in the lifestage that are available for movement to other subpopulations via the dispersal process. For example, if individuals in a particular stage disperse only when they are less than 5 days old, the process could be a function of **Chronological Age**, with the function becoming 0 for cohorts with a **Chronological Age** at and above 5. Note that this process must be used in those lifestages where the dispersal process is used or no dispersal will take place.

## 5.2.4 Dispersal

**Dispersal** is the process that actually moves individuals between different spatial subpopulations. In most ways, it acts in a way very similar to the Stage Transfer process, with the dispersal rate determining the proportion of individuals moving. However, because it is specified by properties of both subpopulations (cells) of each pair of cells involved in the dispersal (the originating and destination cells), a slightly modified syntax is used to describe dispersal. Please refer to the ***Builder Users Guide*** for a full description of the dispersal process and how it is specified.

If  $n$  subpopulations are defined for a model, then there will be  $(n-1)$  dispersal paths leading out of each cell, each with its own dispersal rate. How many individuals actually disperse to each of the other cells on a particular time step will depend on the value of each dispersal process and these are calculated as follows. Assume  $d_{i,j}$  is the calculated dispersal rate (as a proportion of individuals in the cohort) for a particular cohort moving from cell  $i$  to cell  $j$  via the dispersal process. The proportion of individuals actually dispersing from the cohort on that timestep ( $D_i$ ) is then calculated as follows ( $\prod$  is a “product” operator)

$$D_i = 1 - \prod_{k=1; k \neq i}^{k=n} (1 - d_{i,k})$$

Then the actual proportion of individuals dispersing from cell  $i$  to cell  $j$  ( $D_{i,j}$ ) is

$$D_{i,j} = D_i \times \frac{d_{i,j}}{\sum_{k=1; k \neq i}^{k=n} d_{i,k}}$$

### 5.2.5 Genetic Mixing

**Genetic Mixing** is the process that mixes the genotypes in a lifecycle that uses genetic subpopulations. In the current version of CLIMEX, this is restricted to the situation of a single, autosomal locus with two alleles with Mendelian inheritance, so that each dimension component represents one genotype (AA, Aa and aa). If such a subpopulation structure is specified, the **Genetic Mixing** process is automatically invoked during reproduction in any reproductive lifestage. It assumes random mating across the subpopulations.

### 5.2.6 Stage Transfer



The **Stage Transfer** process moves individuals from one lifestage to the next. The process rate is the proportion moved during any timestep. If the process rate equals or exceeds 1.0 during a particular timestep, all members of the cohort are transferred to the next lifestage. The Product combination rule is generally used for transfer process components.

If a lifestage has two Stage Links exiting from it, the number of individuals actually transferring to each of the next stages on a particular time step will depend on the value of each process rate and are calculated as follows. Assume  $t_1$  and  $t_2$  are the transfer rates for branch 1 and branch 2, respectively, from a particular cohort. Then the proportion of individuals *graduating from the cohort* on that timestep ( $p_t$ ) is

$$p_t = 1 - (1 - t_1) \times (1 - t_2)$$

The proportion of individuals transferring to the next stages via each branch is:

$$\text{branch 1,} \quad p_1 = p_t \times \frac{t_1}{(t_1 + t_2)}$$

$$\text{branch 2,} \quad p_2 = p_t \times \frac{t_2}{(t_1 + t_2)}$$

## 5.2.7 Fecundity and Progeny Production



**Fecundity** and **Progeny production** are processes that appear only in a reproductive lifestage. **Fecundity** can be either an *establishment* process (i.e., it is calculated only when a cohort is created) or a *continuous* process. The former is used to set the initial value of the **Residual Fecundity** Cohort Property. The latter “recharges” the Residual Fecundity at intervals. The **Progeny production** process calculates the number of new individuals produced every timestep through reproduction. Progeny Production is very similar to a Transfer Process. The current value of Residual Fecundity sets an upper limit to the progeny production at any time. The Residual Fecundity is decremented by the number of progeny produced at each timestep. The progeny appear in the destination lifestage at the beginning of the next timestep. One difference between Stage Transfer and Progeny Production is that the former is always scaled between 0 and 1 (i.e., a proportion of the number of individuals in the stage), while the latter specifies the actual number of progeny produced.

## 5.3 Dummy Cohorts

Dummy cohorts are used in a DYMEX simulation to enable calculation of variables such as Development Time even when no individuals of the lifestage under consideration are present in the simulation. For example, after running a model that predicts insect development, we may require a graph of time from egg-lay to hatch (egg development time). If no egg laying takes place during a particular period, that period would show up in the graph as a blank (i.e., since no egg cohorts were formed, no development times could be calculated). One possible way to overcome this may be to initialize the egg stage with a small number of eggs during that period, but those eggs would then become part of the simulation, which may be undesirable for other reasons.

DYMEX can add cohorts, referred to as *Dummy Cohorts*, to allow the calculation of development in the above case at any time when normal cohorts of the lifestage are not present. These Dummy Cohorts are treated the same as any other cohorts for all purposes, except that they are considered to have a Number of 0 – i.e., they contain no individuals and do not therefore add to the model population. The addition of Dummy Cohorts is controlled from the Cohort Removal dialog (see *Cohort Removal*, page 42).

Dummy Cohorts are initialized using a Start State (Fig. 5-6) with the name of “Dummy”. If such a start state has not been defined, the “default” start state (consisting of default cohort variable initialization settings) is used instead. In order for the calculated development times to be correct, it is important to consider creating an appropriate “Dummy” start state.

## 5.4 Lifecycle Factors

A number of Lifecycle Factors can be specified for a lifecycle. As with factors associated with other DYMEX modules, each of these can be a parameter, function or process. Each of the factors becomes a local variable in the

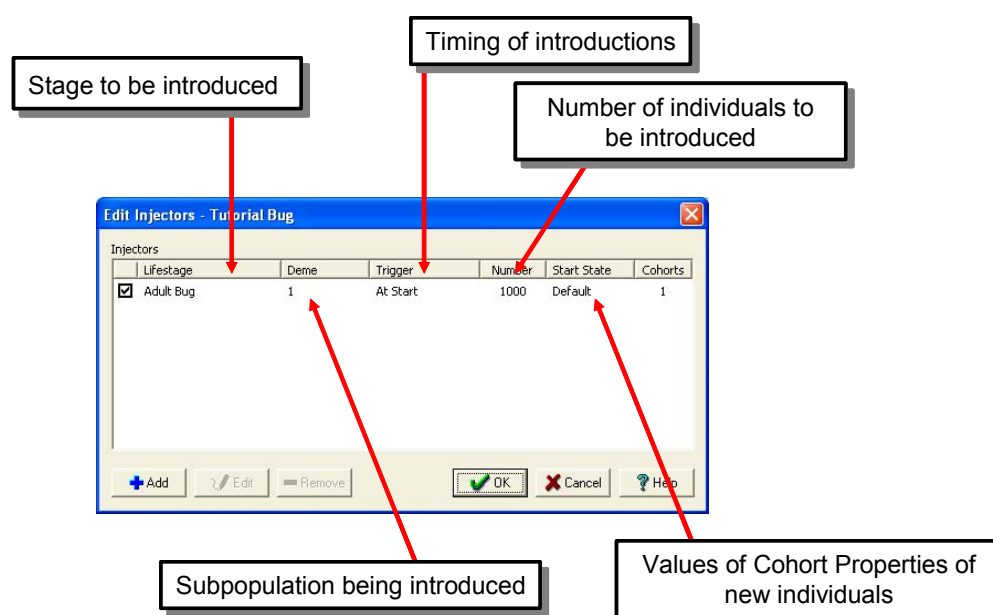
lifecycle and can be used as driving variables for lifestage processes (Section 5.2). The facility is useful for specifying parameters and variables such as *Sex Ratio* which may be used in several lifestage processes in more than one lifestage. They can then be defined just once as a Lifecycle Factor and used wherever necessary.

## 5.5 Lifecycle Initialization

Initializing a lifecycle consists of introducing individuals into the population during the simulation. Individuals can always be introduced at the start of a simulation. If the lifecycle input variables have been linked to the **Timer** by the model designer, then individuals can also be introduced at other times. Repeated introductions of individuals can also be performed.

An *Initialization Set* (or *Injector*) is a set of related introductions of individuals into the simulation, specifying the stage involved, the number of individuals to be introduced each time and the timing of the introductions. Such a set can consist of a single introduction (for example, 10000 larvae on 15 March 1975), or a related series of introductions (for example, 50 adults on 21 July 1990, and every seven days thereafter for a year). As many initialization sets as required may be specified for a simulation run.

Fig. 5-2 Lifecycle initialization dialog box.



### ➤ To initialize a lifestage

1. Left click on the module and select **Initialize Module**. The **Edit Injectors** dialog will appear. Any initialization sets already created for this lifecycle are listed in the "Injectors" panel (Fig. 5-2).
2. Left click on the **Add** button to create a new initialization set (injector), or select an existing set by clicking on it and then the **Edit** button to modify it.



3. An initialization set can be disabled by un-ticking the box in front of its name.

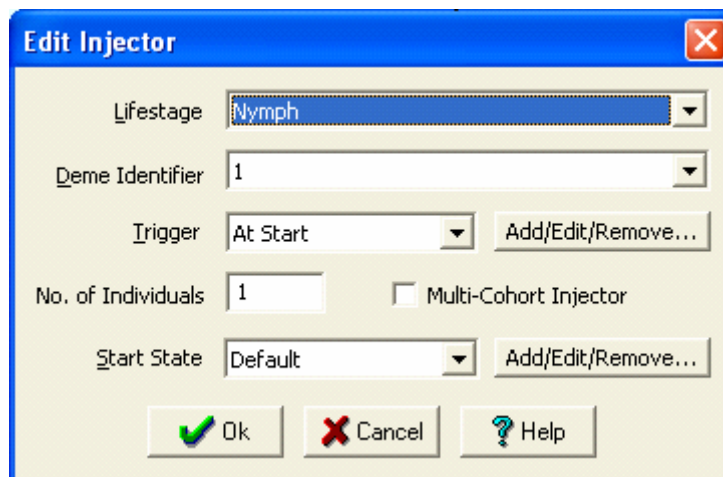
When either the **Add or Edit** button is clicked, the **Edit Injector** dialog box appears (Fig. 5-3). This allows the lifestage as well as the number of individuals to be added to be specified. It also allows selection of the “Trigger” that specifies the timing of the introductions. Finally, a “Start State” (which specifies a value for each user-defined Cohort Variable) can be selected to allow the introduced individuals to have a given state.

4. Select the lifestage of the individuals to be added.

Note that the model designer may have prohibited the user from initialising one or more lifestage. In that case, those lifestages will be missing from the list of lifestages.

If subpopulations are used in the model, the subpopulation being used for this injector can be selected from the drop-down list labelled **Deme Identifier**. If subpopulations are not used, this selection will be unavailable.

Fig. 5-3 Edit Injector dialog box.



5. Select the “Trigger” to be used to time the introductions. The “AtStart” trigger is always available, and adds the individuals at the beginning of the simulation. Other triggers can be defined by clicking on the **Add/Edit/Remove** button next to the trigger selection box (see below).
6. Enter the number of individuals to be added into the lifestage at each introduction.
7. Select the “**Start State**” of the individuals to be introduced (see below for more).

Normally, a single cohort is created containing the number of individuals specified. However, in some rare situations, it may be useful to create a

separate cohort for each individual. Where that behaviour is required, the **Multi-Cohort Injector** check box must be checked.

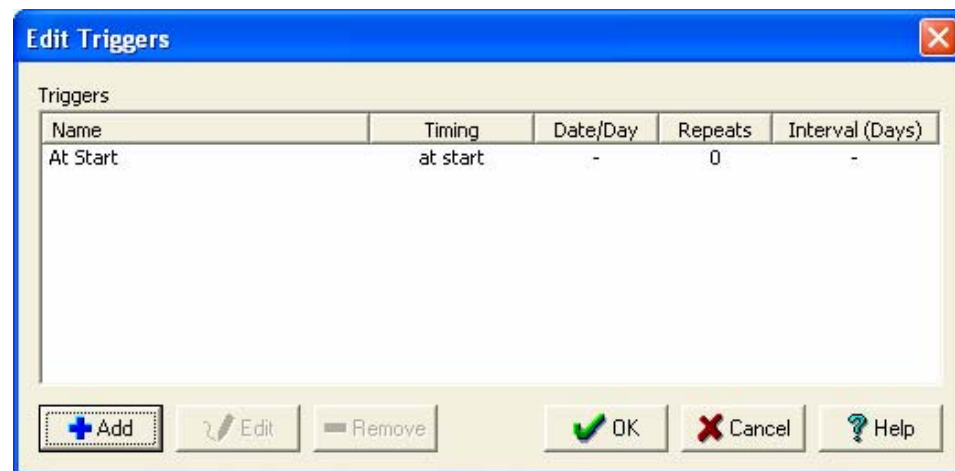
➤ **To initialize or edit a trigger**

1. Click on the **Add/Edit/Remove** button beside the box showing the trigger selection. This will display the **Edit Triggers** dialog (Fig. 5-4), which gives the list of triggers that have been defined for the lifecycle.
2. Left click on the **Add** button to create a new trigger, or select an existing set by clicking on it and then the **Edit** button to modify it. This opens the **Edit Trigger** dialog (Fig. 5-5).



*Note that the dialog will appear differently, depending on whether or not certain Timer variables have been linked to the Lifecycle inputs. Thus, for some models, either or both the “On Date” or “On Day (of Year)” options may be unavailable. See the Builder Guide for more information on linking the lifecycle to the Timer module.*

**Fig. 5-4** Edit Triggers dialog box, listing the initialization triggers defined for a lifecycle.



3. Select the timing required for this trigger by choosing either **At Start**, **On Date** or **On Day (of Year)**.
4. Specify the number of repetitions required and the interval between repetitions.
5. Supply a name for the trigger. This name will be listed in the trigger selection window of the **Edit Injector** dialog (Fig. 5-3).

Repeated Introductions

Repeated introduction of individuals into the population can be performed in two ways. Setting an introduction on **Day (of Year)** adds the specified number to the population on that day in each year of the simulation. Setting one introduction at the **Start** of the simulation or on a particular date and then setting a number of **Repetitions** adds the specified number of individuals on

the given day and on subsequent days as indicated by the interval and repetitions. The **Interval (days)** must be set to equal the number of days between **Repetitions** of the introductions.

**Fig. 5-5** The “Edit Trigger” dialog box, which allows a trigger to be edited or created.

Normally, individuals are added to a lifecycle (within their cohorts) with the Cohort Variables set to their starting values, as defined by the model builder (refer to the *Builder User's Guide*). This is the “default” *Start State*. Sometimes it is useful to add individuals whose properties differ from those starting values. This may occur, for example, where it is required to simulate a particular experiment where individuals of a known age structure are added to a population. To do this, DYMEX allows the user to define other start states which can then be used for the initializations.

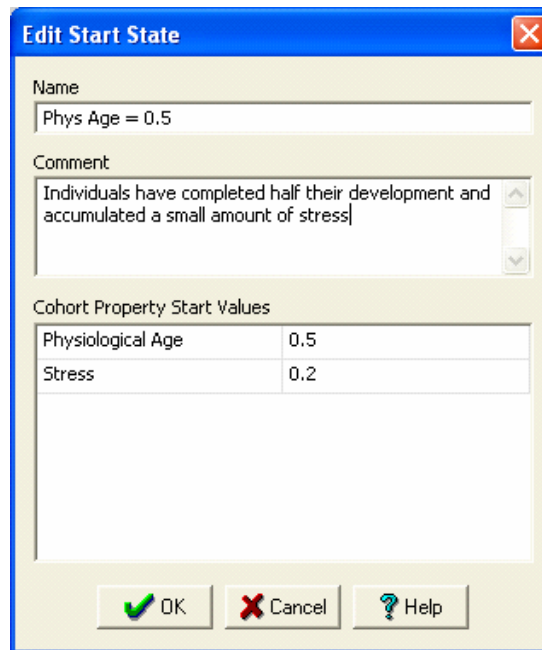
➤ **To initialize or edit a “start state”**

- 1.** Click on the **Add/Edit/Remove** button beside the box showing the start state selection. A dialog that lists the currently available start states will be shown.
- 2.** Left click on the **Add** button to open the **Edit Start State** dialog (Fig. 5-6).
- 3.** Fill in the Cohort Property Start Values required for the start state. An optional comment may be added. A descriptive name must be supplied, it will be the name that appears in the Start State selection box in the Edit Injector dialog (Fig. 5-3).

Editing and  
Deleting  
Introductions

Any *Initialization Set* (Injector) can be edited at any time by selecting the appropriate set and then left clicking on **Edit**. An initialization set can be removed completely by selecting it and left clicking on the **Remove** button.

**Fig. 5-6 The “Edit Start State” dialog box, which allows starting values for Cohort Variables to be specified for lifestage initializations.**



## 5.6 The Lifecycle Diagram

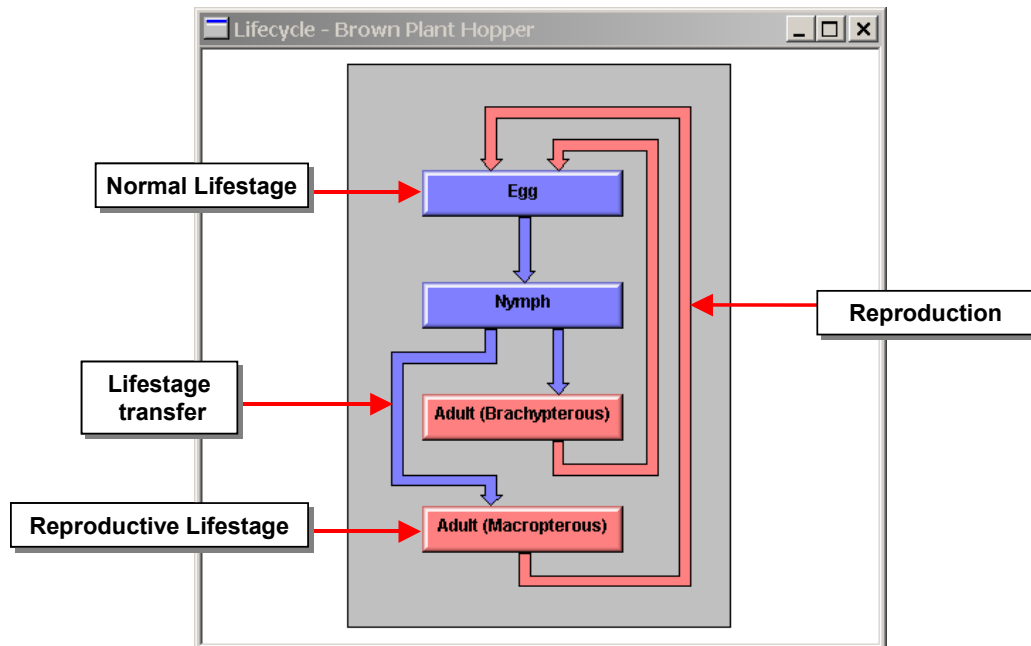


Choosing **Show Lifecycle Diagram** from the Lifecycle menu displays a simple diagram of the lifecycle structure in a Lifecycle window ( Fig. 5-7). To remove the lifecycle diagram, close the window using the close button at its top right corner.

The purposes of the diagram are to illustrate how the lifestages are connected and to give access to a graphical representation of the processes within each lifestage. Each rectangular box represents a lifestage, with those shaded blue being non-reproductive lifestages and those in red representing reproductive lifestages. *Endostages* are shown as light-blue boxes placed within the containing lifestage. The blue arrows indicate stage transfer, while red arrows are used to depict the movement of progeny from the reproductive (parent) stage to the destination stage. Note that endostages are always contained by a reproductive stage and the “progeny” arrow is not shown in that situation.

When the Lifecycle Window is active, a Lifecycle menu item is present on the menu bar. The user can change the current graphics mode from this window (which changes the actions in response to mouse clicks). The operations available from the Lifecycle menu (also available from the tool bar) are as follows:

Fig. 5-7 Example Lifecycle Window showing a branching lifecycle.



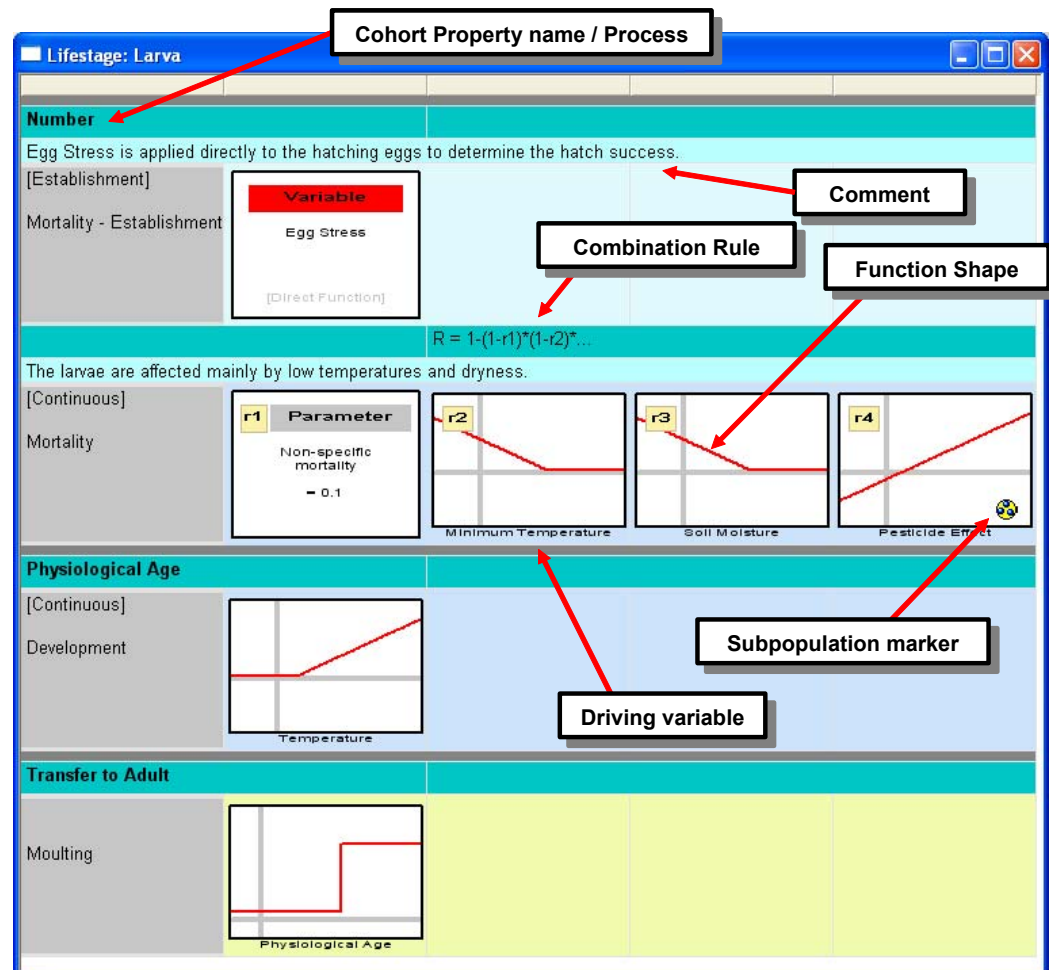
- **Query Mode** – places the Lifecycle Window into the “Query” mode. This mode is indicated by a cursor with a pointer and question mark (🖱️?). When the window is in this mode, and the left mouse button is clicked when the cursor is over a lifestage box, the corresponding Lifestage Window is opened (see next section).
- **Pan Mode** – places the Map Window into the “pan” mode and is indicated by the pan cursor (🖱️). In this mode, with the mouse button down, the lifecycle diagram can be moved within the window.
- **Zoom Mode** – places the Lifecycle Window into “zoom” mode, indicated by the zoom cursor (🖱️). This then allows more detailed examination of any part of the lifecycle diagram. Two methods can be used to zoom in onto a point or region. Left-clicking on a point in zoom mode causes an enlarged version of the diagram to be drawn (the enlargement is 50%), positioned so that the point that was clicked stays in the same absolute position on the screen. Similarly, right-clicking on a point draws a 50% smaller version of the diagram, again positioned so that the clicked point does not move. There are limits to the enlargement and reduction allowed. An alternative way to zoom into an area is to left-click on the point that is required as the top-left corner of the zoomed area and, leaving the mouse button down, move the mouse to the desired lower-right corner of the area. A rectangle outlining the zoomed area will appear as the mouse is moved on the screen. When the mouse button is released, the area outlined by the zoom-rectangle will be drawn to occupy the full size of the window.
- **Show all** – without changing the display mode, redraws the lifecycle diagram at the maximum size to fit wholly within the window.

The Lifecycle diagram can be printed from the **File|Print** menu (with **Print Preview** also available).

## 5.7 The Lifestage Window

The Lifestage window for a particular lifestage can be accessed in two ways, (1) clicking on the selected lifestage box in the Lifecycle Diagram (see Section 5.6), or (2) selecting the **Lifestage** option from the **View** menu and then clicking on the name of the required lifestage in the popup menu. If more than one **Lifecycle** is present in the model, a popup menu will appear before the lifestage selection to allow selection of the lifecycle. The Lifestage window contains a set of graphics each of which shows a Process Component, grouped into Processes (Fig. 5-8).

Fig. 5-8 An example of a lifestage window.



A broad, dark-green band, labelled with the name of the Cohort Variable or process heads each type of process. Within each process region, the process factors are depicted by graphs showing the function shape and driving variable for the factor. For those process factors that are parameters, a box showing the parameter name and its current value replaces the graph. If a "Direct" function has been selected for a factor (i.e., the value of the Driving Variable is used

directly as the factor value), a box labelled “Variable” and showing the name of the driving variable is displayed.

In the example illustrated in Fig. 5-8, two mortality processes are shown. One (labelled “Mortality - Establishment”) is an *establishment* process, and uses the value of **Egg Stress** to remove a proportion of the hatching eggs as they enter the Larval Stage. The other (labelled “Mortality”) is a *continuous* process, and imposes mortality on the larvae when it is either too dry or too cold. A constant mortality factor is also included (as a parameter), and a fourth factor accounts for mortality from a pesticide treatment. Note the subpopulation marker that is shown in this process factor – it indicates that the factor can take different values for different subpopulations. The “Development” process (affecting **Physiological Age**) is driven by **Temperature**, with the increment in Physiological Age per timestep increasing linearly above a threshold temperature. The Physiological Age is used to drive Stage Transition (i.e. “Moulting”) using a “Step” function.

Moving the left mouse button over a process factor graphic will display any description that may be associated with that factor. Clicking the left mouse button when the cursor is located over a process factor graphic results in a dialog that describes the factor in more detail and lists its parameters. The exact form of the factor’s functional relationship can also be obtained from this dialog. The user can change the value of any of the factor’s adjustable parameters by clicking on its name and then providing a new value.

To close the Lifestage window left click on the close window button in the top right of the window.

## 5.8 Examining the Cohort Properties



Selecting **Show Cohort Properties** from the Lifecycle popup menu will display a list of all the Cohort Properties for that lifecycle, along with their scope, update methods and initial values (Fig. 5-9). Any of the Cohort properties (listed in the panel at the top of the dialog) can be selected. The panels at the bottom then show a description of that cohort property (if one has been supplied by the model designer) and the stage links in which it is reset. In the example shown in Fig. 5-9, the *Basal Area (cm<sup>2</sup>)* cohort property is reset as individuals pass from the “Adult” stage to the stage named “Buds-Imm. Pods/Seeds (tree)”. The thin arrow (→) between the names of the lifestages indicates that this link is a reproductive link. Thick arrows (➔) are used to indicate non-reproductive links.

Four cohort properties (*Number*, *Density*, *Chronological Age* and *Physiological Age*) are present in every model (though they are not necessarily used), while others will be present only if the model designer has created them for use in this lifecycle.

**Scope** The **Scope** determines whether the corresponding cohort property keeps its value between lifestages. A local property does not retain its value between lifestages (e.g. *Chronological Age* is always set to zero when a cohort is

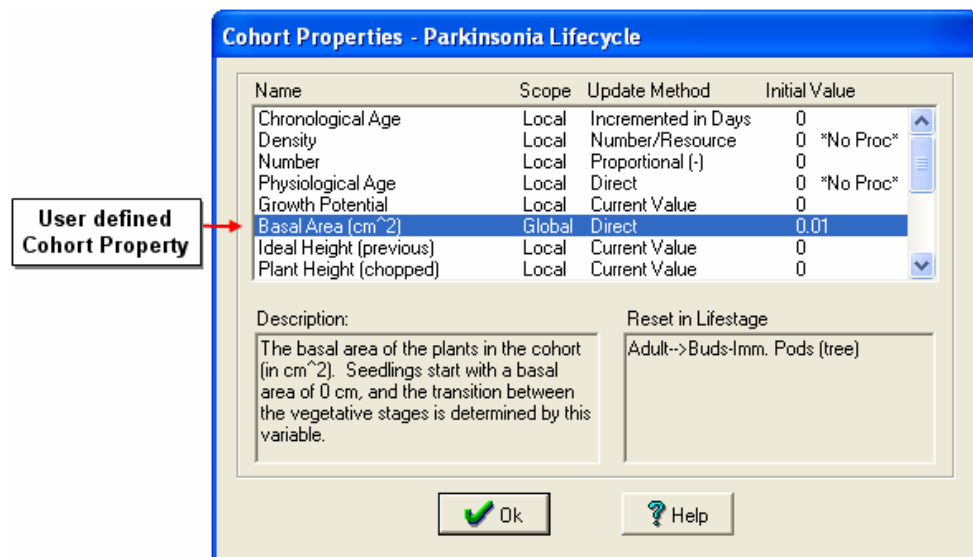


created). Global Cohort properties maintain their value between lifestages (except where a reset to their initial value has been specified by the model designer) and are used to carry information between lifestages.

#### Update Method

The **Update Method** determines how the cohort property is updated. The available update methods are described in *Lifestage processes and their characteristics* on page 29. If the update method is followed by “(-)”, the updating is inverted (i.e., rates are subtracted rather than added). An asterisk just in front of the update method indicates that this is a cohort property which uses the “most recent” values of the driving variables in its updating processes (rather than the values from the previous timestep). This is discussed in more detail in Section 6.14 of the Builder User’s Guide). Note the special update methods used by *Chronological Age* (which is incremented either in days or timesteps – set by the model designer) and *Density* (which is always equal to the *Number* divided by the value of the **Resource Variable**).

Fig. 5-9 An example section of the Cohort Information dialog box.



#### Initial Value

Local Cohort Properties are initialized to the **Initial Value** when they are created. Global properties are only set to this value in cohorts created after the stage in which a reset is specified. For example, ‘Stress’ has an initial value of 0 in the example above, as that is the value we want ‘Stress’ to start with when it is not being transferred from the previous stage. The phrase “\*No Proc\*” indicates that the Cohort Variable has no process updating its value in any lifestage. Note that this does not mean it is not used as an input variable somewhere, but that its value will not change from its initial value.

#### Multiple Lifecycles

If the model contains more than one lifecycle, **Show Cohort Properties** for the lifecycle of interest will only show the cohort properties for that lifecycle.

## 5.9 Cohort Removal

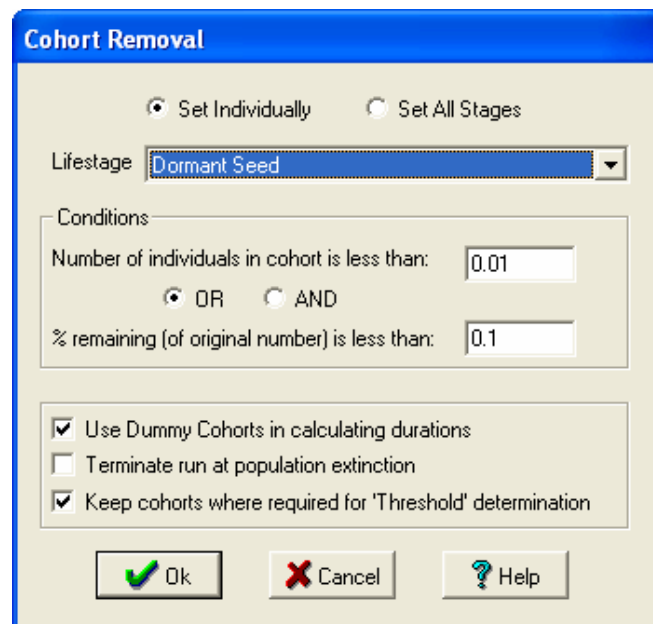
When the population of a cohort becomes so small that the number is ‘insignificant’, the cohort can be removed from the simulation. Leaving the



cohort in the simulation could result in numerous very small cohorts that may slow the simulation considerably.

The **Cohort Removal** dialog box (Fig. 5-10) is accessed from the **Execution** menu by choosing the **Cohort Removal** option. If the model has more than one lifecycle module, a popup menu allows selection of the required lifecycle. The conditions under which a cohort is removed from the simulation (the *removal conditions*) can be set individually for each lifestage (**Set Individually**) or the same removal conditions can be set for all lifestages (**Set All Stages**) by clicking on the corresponding button. If the former option is selected, a lifestage must be selected from the **Lifestage** list (by clicking on the small “drop-down” button to its right). Two separate conditions for removal must be specified: the actual number of individuals in the cohort below which cohort removal can take place, and the percentage remaining (of the original number of individuals when the cohort was formed) below which cohort removal takes place. Using the two buttons (OR and AND), the user can specify whether both conditions must be met for cohort removal (AND) or either condition suffices (OR). The latter is used by default.

**Fig. 5-10 Cohort removal dialog box with default settings.**



For example, if a cohort that was formed with 10 individuals and the default values (0.01 individuals OR 0.1%) are used for cohort removal, the cohort will be removed if it contains fewer than 0.01 individuals. *Note because DYMEX models populations and not individual organisms it is possible to have 0.01 of an individual in a cohort (i.e. in other words a density).*

Different cohort removal settings can be set for each lifestage. Setting the values higher can speed up the simulation, but caution is needed to ensure that the values are not set so high that they distort the results.

**!** If stage mortality is being output from the lifestage as a variable, and the population consists of many small cohorts, the cohorts will be removed and

incorrect stage mortalities may be calculated. To avoid this problem set the cohort removal conditions to very low values.



Caution is also needed if “genetic” subpopulation are used to, for example, examine the development of resistance under different chemical treatment regimes. In such a scenario, the simulation would usually be initialized with a very small proportion of resistant or heterozygous individuals. The wrong settings of cohort removal conditions could then severely distort the results by removing a larger proportion of resistant individuals. For that reason, the default values are automatically set to 1.0e-8 individuals OR 0.1% when genetic subpopulations are used by the lifecycle.

**Dummy Cohorts** Three other options are available from the **Cohort Removal** dialog. The **Use “Dummy Cohorts” in calculating lifestage durations** option, when checked, enables the creation of Dummy Cohorts for the lifecycle. Note that Dummy Cohorts are either enabled or disabled for the whole lifecycle (see *Dummy Cohorts*, page 33).

If the **Terminate run at population extinction** is checked, the simulation will terminate as soon as there are no more cohorts (excluding Dummy cohorts) of the corresponding lifecycle in the simulation. This can speed up running of the model considerably, especially in multiple runs. However, caution is needed when using this option in conjunction with Lifecycle initialization during the simulation, as it may cause the run to terminate before the initialization has taken effect.

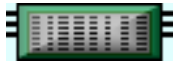
The last option (**Keep cohorts where required for ‘Threshold’ determination**), when selected, keeps cohorts that would otherwise satisfy the cohort removal criteria if they are needed for calculating a ‘threshold’ output. An example of such a threshold output is “Time until Physiological Age reaches 1”.

## 5.10 Lifecycle Module Run Options

These options are accessed from the Model Components Window by clicking on the appropriate Lifecycle graphic and then selecting “Module Run Options” from the resulting popup menu. In the current version of DYMEX, just a single option is implemented. When selected, it provides a warning message whenever a lifestage process attempts to move its associated Cohort Variable outside its valid range. This could happen, for example, if a mortality process evaluates to a number above 1, which would attempt to make the *Number* Cohort Variable become negative. For most purposes, a warning for this condition is not required, as DYMEX will always keep the variable within the allowed range defined in the *Builder*. However, under some circumstances, the knowledge that a process is attempting to move a particular variable outside its allowed range can help in finding problems with a misbehaving model.

☒ Warn when a process attempts to move Cohort Variable out of range.

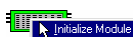
## 6 The DataFile and MetBase modules



The **DataFile** module allows the model to read in a series of values from a file at each timestep during a simulation. The model designer sets the number of variables that are to be read from the file as well as the variables' names. More than one **DataFile** module may be present in a model. The **MetBase** module is a specialized type of **DataFile** module, with the feature that it has a number of pre-set variables (Daily Maximum Temperature, Daily Minimum Temperature, Rainfall, 9am Relative Humidity, 3pm Relative Humidity and Evaporation). Some of these pre-set outputs might not be used in a particular model, while additional output variables may be present in some models. Initialization of the module is almost identical to **DataFile** initialization. In this section, unless stated otherwise, any reference to **DataFile** will also include the **MetBase** module.

Input variable	If the <i>Simulation Date</i> variable is linked to the module's input, the sequence of dates in the associated file will be checked during a simulation run.
Model Timesteps and Data Timesteps	There is no requirement in DYMEX that the timestep of the model (set in the <b>Timer</b> module) and the timestep of data files must be the same. Both the <b>DataFile</b> and <b>MetBase</b> modules will interpolate or average values as required if these timesteps are different.
Initialization	The output variables from this module correspond to the data items to be read from the associated file. Initialization consists of selecting the actual file that the data is to be read from, and associating columns within that file with the corresponding module output variables. In addition, various options that define how missing data is treated can also be set.

### 6.1 Metbase & Data File Initialization



Initialization of the data input modules is often the most complex part of initializing DYMEX. Two types of formats can be used with these modules, "fixed-width" or "comma-delimited". Both data formats require that the data for each data timestep is present on one line of data, with lines separated by "newline" characters. A maximum of 99 lines can precede the data, with this information being ignored by the reader.

The "fixed-width" format requires that each data item in the file occurs in the same position on each line. For example, in a meteorological data file, the minimum temperature may take up characters 12 to 16, inclusive, on each line. This corresponds to a typical FORTRAN file format for users that are familiar with that language.

The "comma-delimited" format separates data items on each line by a "," character. With this format it is merely required that each data item takes up the same relative position on a line, in terms of commas. For example, the minimum temperature could always be the 3<sup>rd</sup> item (i.e., follow the 2<sup>nd</sup> comma).

## 6.1.1 Importing Spreadsheet Files into DYMEX

The comma-delimited format is the most convenient when the data originates from a spreadsheet such as MS Excel. To save data as comma-delimited files in Excel go to **Save File As...** and choose the comma delimited format. Other spreadsheets will have similar ways of saving a comma-delimited file. Then use that file as input to the required module.

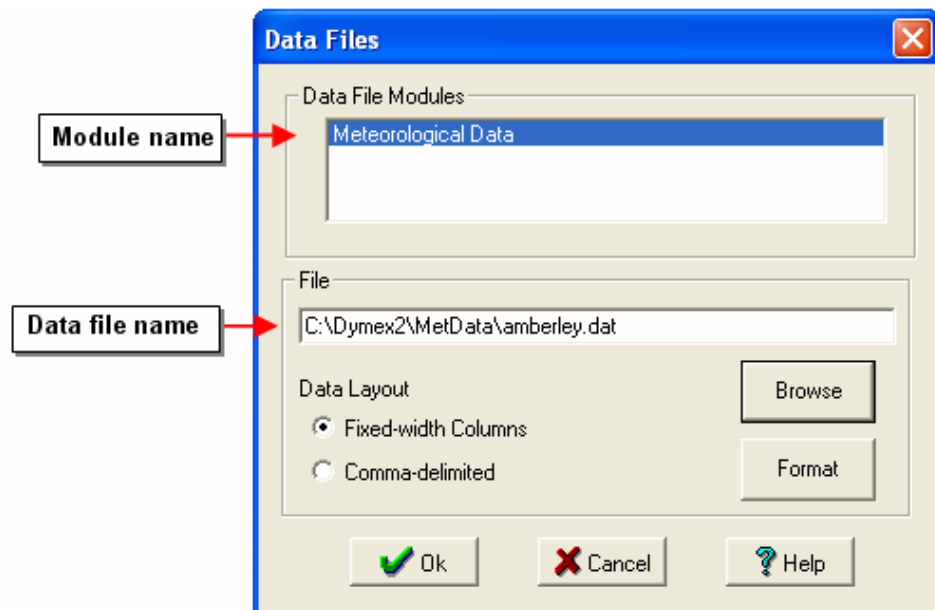
## 6.1.2 Initializing Data File Modules (Basic Options)

### ➤ To initialize fixed-width column files

#### 1. Left click on the module and select **Initialize Module**.

! This will display the **Data Files** dialog box that allows the user to set the filename of the data file and to format the variables. Both must be done before a module is successfully initialized.

Fig. 6-1 Data File dialog box with example data file.



#### 2. Ensure that the **Data Layout** selection is set to “Fixed-width Columns”.

The Filename **3.** The filename of the data file being used may either be typed in the **Name** box or can be looked for by left clicking the **Browse** button.

The file name dialog box, which appears when the Browse button is pressed, has a filter on it that filters for files ending with the extensions, **dat**, **prn** and **txt**, (or **csv** for comma-delimited files). Files with different extensions can be selected by choosing the All Files (\*.\*) filter from the File Types drop down list.

4. Once the filename has been selected, left click the **Format** button.

This will display the **File Format** dialog box. The **File** window in this dialog displays the first few lines of the file. This window allows the variables (displayed in the **Variables** list box) to be linked to columns of data in the file.

5. To link a variable to a column, first select the required variable by left-clicking on the variable's name within the **Variables** list box and then drag the mouse over the corresponding column of data (left to right or vice versa).

Negative  
values



For example in Fig. 6-2, "*Minimum Temperature*" has been selected and the mouse was dragged from just in front of the column to its right hand side. The reason for the gap in front is that there may be negative temperatures present in the file.

6. Repeat step 5 for all variables.

7. The settings should be checked for correctness by clicking on each variable name in turn in the **Variables** list-box, and verifying that the highlighted portion of the file corresponds to the data for that variable.

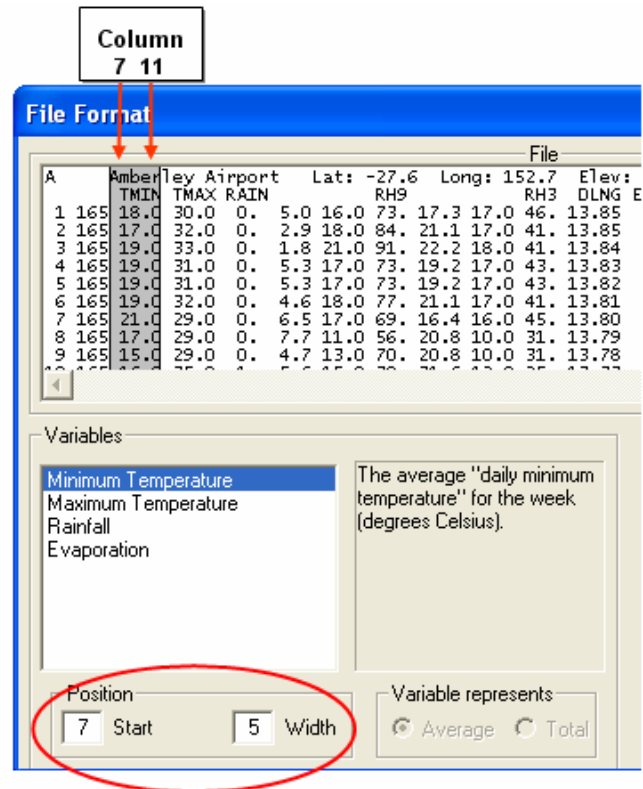
Not all the file  
can be seen!

The **File** window is rather restricted in size (it displays about 12 lines of 120 characters), but can be scrolled so that the first 50 lines of the file can be seen. If the file is wider than what can be fitted in the window, the horizontal scroll bar will become available. When creating a data file, it is worth keeping this in mind to make it easier to set up the format in DYMEX. An alternative to using the mouse dragging technique described above would be to enter the start column and width of fields using the **Position** boxes. In a text editor count the columns which store the data, for example in Fig. 6-2 the Minimum Temperature is within columns 7 to 11 (start column 7, width 5), then type in the values in the **Position** boxes.

Variable  
Represents

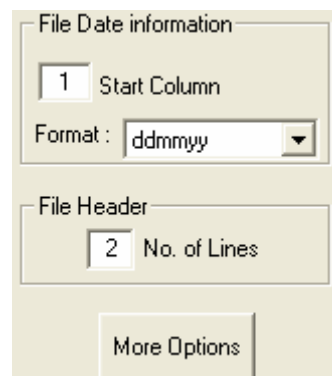
The **Variable represents** setting is important if the timestep of the model differs from the timestep of the data (for example, a daily model with a weekly data file). For example, a maximum temperature generally represents an *average* (in a weekly file it is the average of the 7 daily values), while rainfall is generally recorded as a *total* (in a weekly file, it is the total of the 7 daily values). The **MetBase** module automatically uses the correct settings for the 6 pre-defined meteorological variables. For **DataFile** variables, or additional **MetBase** variables, the user must set the appropriate option for each variable being read by the module to ensure that the data is interpolated or averaged correctly in all cases.

**Fig. 6-2** The top-left section of a Metbase format dialog box showing the selecting of the Minimum Temperature column.



**Date** Once all the variables have been linked to columns in the data file, the position and format of the date information in the file must be specified. In the example file shown in the **File** window of Fig. 6-2, the date starts in the first column and is the format of “ddmmyy”. This information is entered into **File Date information** section of the **File Format** dialog (Fig. 6-3). The **Start Column** is set to the corresponding value (1 in the example), and the required format is selected from the large range of date formats available in the **Format** drop down list.

**Fig. 6-3** The section of a Metbase/ Data File format dialog box relating to date and header information.



**File Header** The file header is the number of lines at the top of the data file that do not contain data. In the example file of Fig. 6-2, there are two lines of header, and

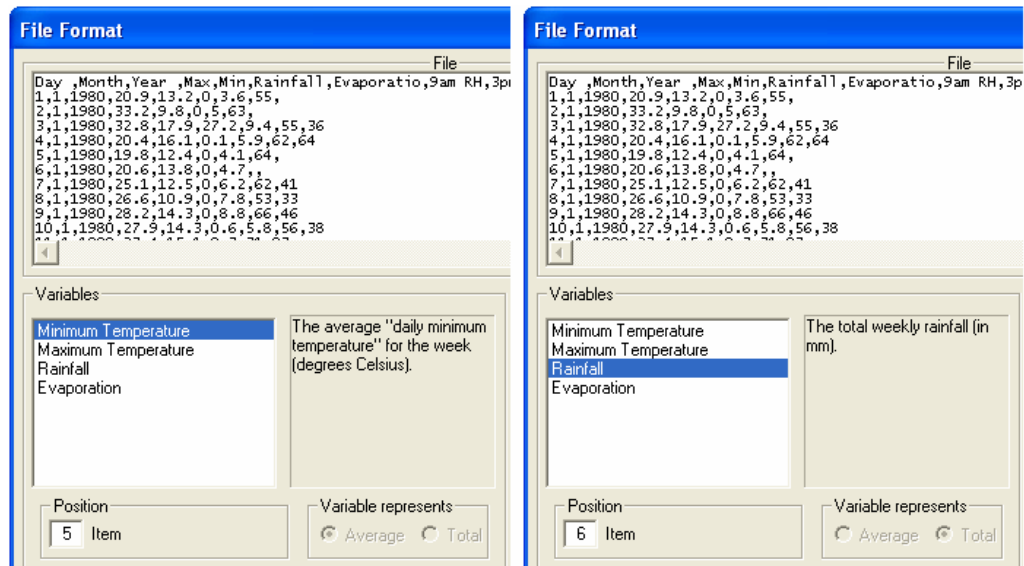
this is typed into the corresponding box (Fig. 6-3).

➤ **To initialize comma-delimited files**

1. Within the dialog box where the name of the meteorological file is selected, the **Data Layout** options allow the selection of the format of the data file. Select the **comma-delimited** format and then left click the **Format** button.
2. Select each variable in turn and type its relative position on the line into the **Item Number** box. It's relative position is effectively equal to one more than the number of commas that precede it on the line.

For example, in the example meteorological data file shown in Fig. 6-4, the *Maximum Temperature* data follows the 3 date columns, so its **Item Number** is 4, the *Minimum Temperature* is the next column and has an **Item Number** of 5, while *Rainfall* has an **Item Number** of 6 (as shown).

**Fig. 6-4** Two variables with their associated item number.



All other format items (such as date and header) for the comma-delimited file are the same as the fixed format meteorological files.

## 6.2 Initializing Data File Modules (More Options)

If the **More Options** button in the **File Format** dialog is left clicked, the dialog extends and more settings become available. These options are ones that tend to be used less often, and include settings that allow the **DataFile** module to handle missing data and files without a date column.

### Missing Values

If the data file contains missing values, they can be handled by setting appropriate options in the **Missing Values** section of the dialog (Fig. 6-5). Sometimes, a zero value is recorded as a blank in a file (this is common with rainfall in fixed-width, columnar files). This is indicated by selecting the corresponding variable in the **Variables** listbox, and checking the **Interpret**

**'blanks' as 0** box. Similarly, the module will interpolate any missing values for a variable if the corresponding **Interpolate missing values** box is checked, and a missing value indicator is provided in the **Missing Value Id.** box. The missing value indicator is a string of up to 5 characters that is used in the file to indicate a missing value (commonly, this will be \*, 9999, or similar). The interpolation algorithm uses simple linear interpolation between the last value before and the first value after the missing items. There is currently a limit of 4 sequential missing values – any more and the reading terminates with an error.

**Fig. 6-5 A section of the More Options extension to the Data file format dialog box dealing with missing values and validity checks**

The screenshot shows two panels. The left panel, titled 'Missing Values', contains three options: 'Interpret 'blanks' as 0' (unchecked), 'Interpolate missing values' (checked), and 'Missing Value Id.' with a text box containing an asterisk (\*). The right panel, titled 'Data validity checks', contains 'Check value in range' (checked), 'Minimum' with a text box containing 0, and 'Maximum' with a text box containing 1000.

#### Data Validity Checks

Data values can be checked as they are being read to ensure that they are within a permitted range. To do this, select the name of the variable to be verified in the **Variables** list-box and then select **Check value in range** within the **Data Validity Checks** box (Fig. 6-5). The minimum and maximum allowed values must then be typed into the **Minimum** and **Maximum** boxes, respectively. The model designer may already have set a permitted range when constructing the model, in which case these minimum and maximum values will be used by default in checking.

**Fig. 6-6 The File Date information format when there is no date column in the data file and the associated File timing characteristics that must be set.**

The top dialog box, 'File Date information', has a 'Start Column' text box with '1', a 'Format' dropdown menu showing '(none)', and a 'File Header' section with 'No. of Lines' text box containing '1'. It includes 'Ok', 'Cancel', and 'Help' buttons. An orange arrow points down to the second dialog box, 'File timing characteristics'. This box has 'Date of first line of' with three text boxes containing '1', '1', and '1977', and 'File Time Step (days)' with a text box containing '1'. It also has a checkbox labeled 'Re-use first year's data for successive years' which is currently unchecked.



If the data file has no date information, the **Format** for the date must be set to “**(none)**”. In that case, it is necessary to supply the date information for that file by filling in the boxes in the **File timing characteristics** part of the dialog (Fig. 6-6) in the **More Options** area. The date that corresponds to the first line of data in the file should be provided in the **Date of first line of Data** boxes, while the data timestep (the number of days between lines of data) must be set in the **File Timestep (days)** box. Currently, the **File Timestep** is limited to 1 or 7 days.

No date  
column

A single year’s data can be read from the data file repeatedly by selecting the **Reuse first year’s data for successive years** option at the lower right side of the **More Options** area. This is particularly useful when a MetBase module is being used to read long-term average data, but the model is being run for a period of more than one year.

## 7 The MetManager module

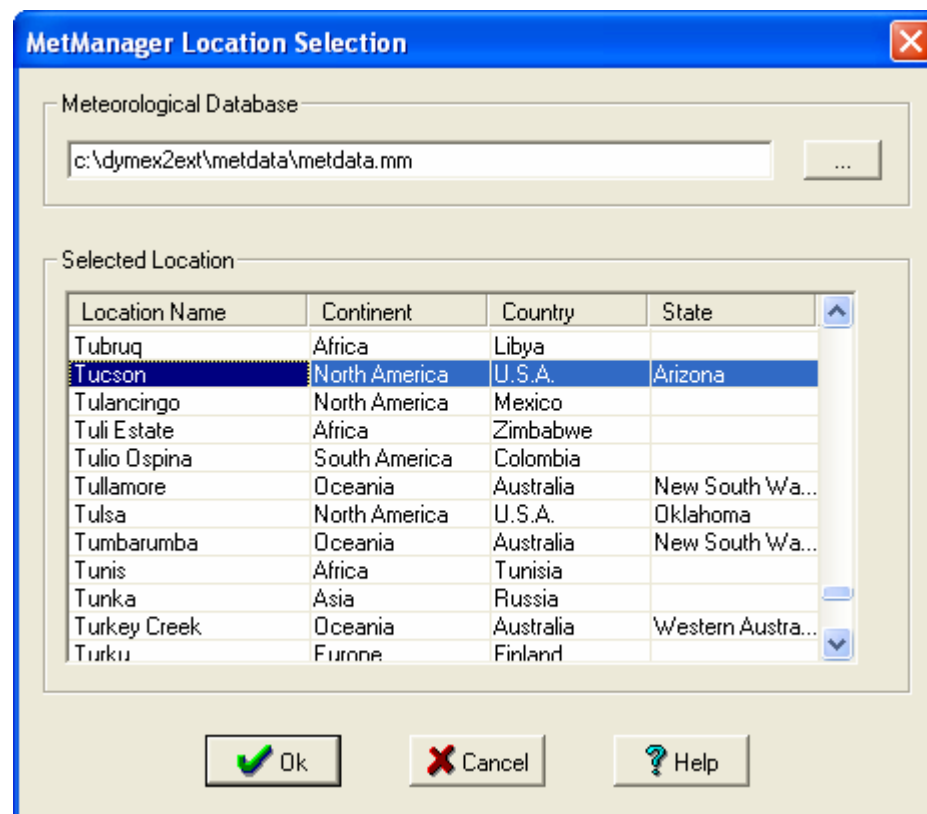


The **MetManager** module is used to provide the values of some climate variables from the “MetManager” database of long-term average climate data. The model designer sets the number of variables that are to be read from the database as well as the variables’ names. Available variables include maximum and minimum daily temperatures, rainfall and 9am and 3pm relative humidity. More than one **MetManager** module may be present in a model.

The climate database as supplied stores monthly long-term averages. When the MetManager module is used in models with daily or weekly timesteps, it interpolates to give daily or weekly values.


The MetManager module can use either an Access® database (with extension “.mm”) created by the **MetManager** application (see Section 30), or it can use a BMT file (with that extension) as used in Version 1 of the **CLIMEX** application.

**Fig. 7-1 The MetManager initialisation dialog.**



### 7.1 Initializing the MetManager module

1. Left click on the module and select **Initialize Module**. This will open the MetManager initialisation dialog (Fig. 7-1).

2. Click on the Browse button () in the **Meteorological Database** panel and select a database file (either an Access® MetManager file with extension .MM or a BMT file). This will fill the list box with the names of all the locations contained in that database file. Note that if the file contains a large number of locations, it can take some time to create the list of locations.
3. In the **Selected Location** panel, scroll down to locate the required location. Location names are listed in alphabetic order by default. So sort the locations in country order, right-click on the heading of the “Country” column. The location names can be resorted in location order by right-clicking on the “Location name” header.



Note that you can quickly select a location by typing its name after the “Selected Location” list is clicked on. Typing a single character will highlight the first location in the list whose name begins with that character.

## 8 The Circadian module



The Circadian module is designed to generate a variable that describes the diurnal change in the value of some quantity. This is particularly useful in situations such as the calculation of temperature-based development rates, where input data is available as minimum and maximum daily temperatures.

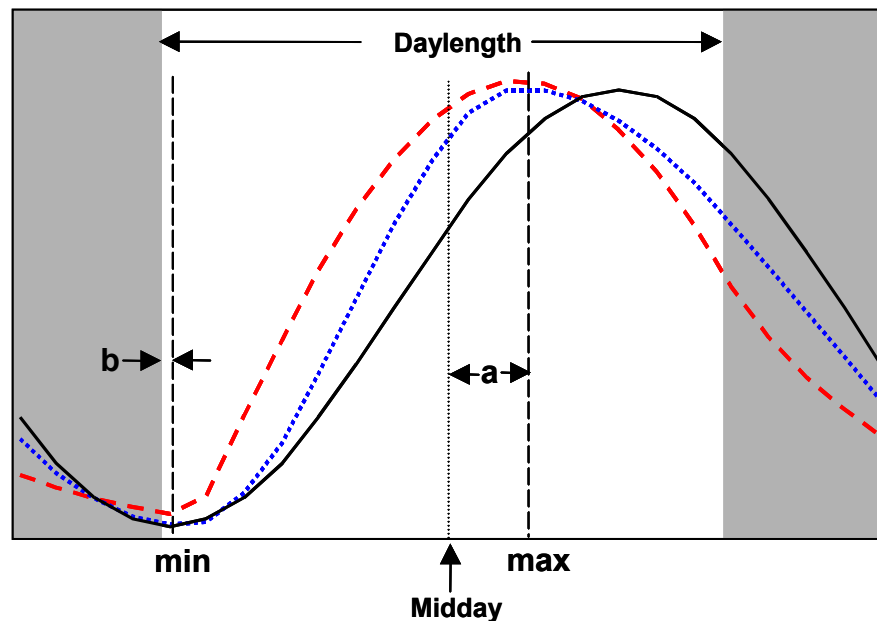
Variables in DYMEX generally are assigned a single value for each timestep. The **Circadian** module output variable is different in that it has a set of values. The day is divided into a number of segments and the output variable has one value for every segment. These values are obtained by interpolation, using one of 3 functions as selected by the model designer. When this output is then used as input to a function, the function is evaluated for each of these values separately, *as if the model had a timestep that is the same as the segment length* (for example, 1 hour). When the output variable is used directly as input to another module, the average value of the Circadian input values is used as input to the module.

A Circadian module will use one of three cycle shapes **Sine**, **Composite (Sine+Sine)** or **Composite (Sine+Exponential)** ( Figure 8-1) to calculate the interpolated values. Both the Composite shapes assume that a minimum occurs before a maximum, and that the maximum occurs before sunset. For the descriptions that follow, the following definitions are assumed:

**a** = the time (in hours) between mid-day (halfway between sunrise,  $t_r$ , and sunset,  $t_s$ ) and the time when the modelled quantity is at its maximum ( $t_{max}$ ).

**b** = the time (in hours) between the time when the modelled quantity is at its minimum ( $t_{min}$ ) and sunrise ( $t_r$ ).

Figure 8-1. Sine ( — ), Composite (Sine+Sine) ( ..... ) and Composite (Sine+Exponential) ( - - ) shapes, in relation to daylength and daily extremes



Daylength is the number of hours between sunrise ( $t_r$ ) and sunset ( $t_s$ ).

- **Sine** is a simple sine curve whose period is 24 hours. Note that the length of the increasing and decreasing parts of the cycle are not affected by daylength, being always each equal to 12 hours. This can give rather poor approximations of the daily change of quantities that are affected by daylength (such as temperature) over the course of a year, especially in locations far from the equator.
- **Composite (Sine+Sine)** is two sine curves joined together at the maximum value. The first curve models the rise in the quantity from the day's minimum value to its maximum, with a rise-time (or half-period) equal to the time between minimum and maximum. The second curve models the fall from maximum to subsequent minimum, with a half-period equal to the interval between these points. The model uses values of 1.86 and  $-0.17$  for **a** and **b**, respectively. See Wann, M., Yan, D. & Gold, H.J. (1985) 'Evaluation of three models of daily cycle of air temperature', *Agric. For. Meteorol.*, 34: 121-128 for more details.
- **Composite (Sine+Exponential)** uses a portion of a sine curve from the first daily minimum to sunset, joined to an exponential decay function from sunset onwards. The sine curve's period is chosen so that its maximum corresponds to the modelled variable's maximum, and its inflection point corresponds to the variable's minimum. The exponential (decay) curve is used to model the fall in value of the quantity being modelled from sunset to sunrise. The model uses values of 1.86 and  $-0.17$  for **a** and **b**, respectively, and 2.2 for the decay coefficient of the exponential portion of the curve. See Parton, W.J. and Logan, J.A. (1981) 'A model for diurnal

variation in soil and temperature', *Agric. Meteorol.*, 23: 205-216, for a detailed description of this model.

## **8.1 Initializing the Circadian module**

In most cases, no initializations are possible or required for the Circadian module. In those cases where the model designer has used to “adjustable” version of the Circadian module (where functions or processes are used to adjust the values of the input variables before the daily cycle values are generated), one or more parameters may be available for adjustment.

## 9 The QueryUser and QueryFile modules



The **QueryUser** and **QueryFile** modules allow one or more user-specified constants to be entered via the keyboard (or from a file in the case of **QueryFile**). These simple modules have no input variables. The model designer generally supplies default values for the constants, but these will usually need to be adjusted to suit a particular situation in the *Simulator*. The **QueryUser** module can have output variables with subpopulation structure.

Note that the output “variables” from these modules are the constants supplied by the user. Since these do not change in value during the course of a simulation run, they are not available for graphing or tabulating in the normal way. In all other ways, however, they behave and are usable otherwise like any normal DYMEX variable. The three different modules under this heading differ as follows:

- **QueryUser** – each of its variables has a valid range that is continuous between the minimum and maximum allowed values.
- **QueryUser/Discrete** – each of its variables has a set of discrete values from which its value must be chosen.
- **QueryFile** – the same as the **QueryUser** module, with the additional ability to read the values from a file.

### 9.1 Initializing the QueryUser module



#### ➤ To initialize the QueryUser module

1. Left click the module representation in the **Model Components** window and select **Initialize Module**. The **Initialize Variables** dialog box will be displayed.

This dialog box (Fig. 9-1) displays a list-box containing all the “variables” available to be initialized and the selected variable’s current value. If variables have different values for different subpopulations, the name of the subpopulation is shown in brackets after the variable name.

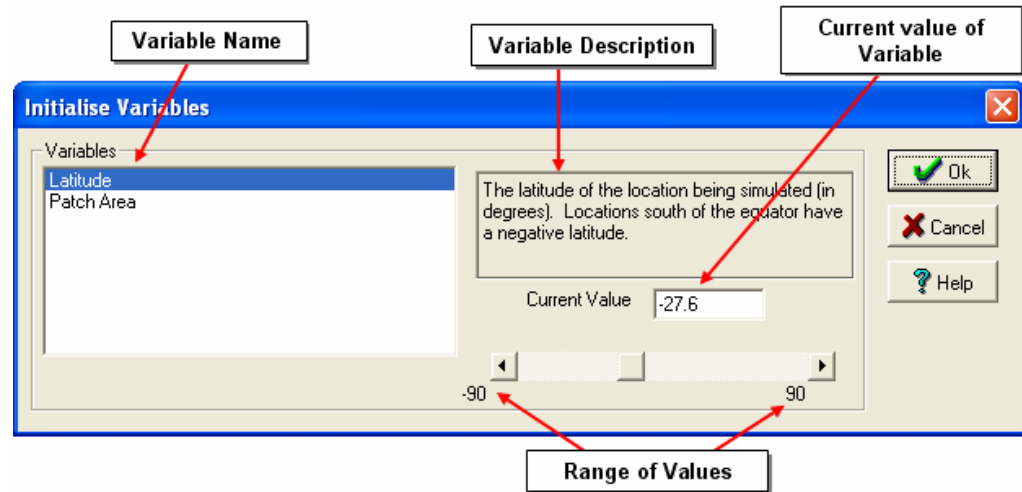
2. Left click on each variable in the **Variables** box to select that variable, and set it to the required value, either by typing the value directly into the **Current Value** box or moving the slider to the appropriate position.

The slider will be disabled if the model designer has not specified the allowed range for this variable. In that case, the value must be typed directly into the **Current Value** box.

Both the slider and the **Current Value** box will be disabled if the model designer has set the minimum and maximum allowed values to be the same. In that case, the variable cannot be changed from its default value.

Note that the values of all the variables will be displayed on the module's display in the Component Window.

Fig. 9-1 Initialize Variables dialog box for a QueryUser Module.



## 9.2 Initialising the QueryUser/Discrete module

The **QueryUser/Discrete** module allows only a discrete set of values to be used for its constants. During model construction in the **Builder**, these values can be associated with names to make the choice easier for the end user. For example, a set of temperatures (say, 10°, 20° and 30°) may have been given the name “Cold”, “Warm” and “Hot”, respectively. By choosing “Hot”, the output variable will be given the corresponding value, 30°.

The initialisation dialog for this module (Fig. 9-2) is very similar to that for the **QueryUser** module. The variable to be set is first selected from the **Variables** list at the left. However, instead of the value being adjusted using an edit box and slider, it is picked from a drop-down list, accessed by clicking the small button (▼) to the right of the box that displays the current selection.

Fig. 9-2 Initializing a QueryUser/Discrete module.



### 9.3 Initialising the QueryFile module



The **QueryFile** module is almost the same as a **QueryUser** module, with the added ability to read constants from a file. To read the constants from a file, the module must be associated with a **DataFile** or **MetBase** module. Thus, for example, the variables “Latitude” and “Longitude” may be read from the same file supplying meteorological data for the corresponding location, so assuring that when the locality is changed by changing to a different meteorological data file, the latitude and longitude information is automatically changed also. If the values are to be read from a data file, the **DataFile** or **MetBase** module associated with that file should be initialized before the **QueryFile** module.

A checkbox in the QueryFile initialization dialog (labelled **Read value for a data file**) is used to indicate that the selected constant is to be read from a file. If that option is not selected, the initialization is exactly as described for the QueryUser module in Section 9.1. If the file option is selected, the initialization dialog expands to add a new section at the bottom (Fig. 9-3). *Note that the dialog will have no checkbox, and behave exactly like a QueryUser initialization dialog, if the model contains no DataFile or MetBase modules.*

Fig. 9-3 Query File module with example variable being read from a file.

**Initialise Variables**

Variables:

- Latitude
- Paddock Size
- Number of Animals
- Season Type
- Nonspecified Dung Removal
- Ground Cover

Latitude of location being run. Note that the latitude of locations south of the equator must be negative.

Current Value: -23.4

☒ Read value from a data file

Data File Modules:

- Meteorological data

Position of Variable in File:

Start Column: 5 End Column: 10

Row: 1

C:\Data\MetData\RepYears\roc5av.prm

Lat	Long	Type	(Av)	GndCov	1.20	Rockhampton (Qld)
1/11/72	33.0	20.5	0	8.0	1972	
2/11/72	35.5	20.5	0	10.0	1972	
3/11/72	37.5	21.5	0	9.8	1972	
4/11/72	35.5	22.0	0	7.6	1972	
5/11/72	33.5	23.5	0	5.2	1972	
6/11/72	33.0	22.0	0	9.8	1972	

In the top left of the lower section of the **Initialize Variables** dialog, there is a list of all the **DataFile** (and **MetBase**) modules that are used within the model. Initialization consists of associating each variable from the **Variables** box that needs to be read from a file with the appropriate **DataFile** module, and specifying the exact position of the data item within the file. If the selected **DataFile** module has already been initialized, the lower window will contain the first few lines of its associated file. Otherwise, it is filled with a block of



numbers that indicate column information. A data item's position within the file can be set by dragging with the mouse in the same way as described for the **DataFile** module, or typing the position of the item directly into the 3 text-boxes labelled **Start Column**, **End Column** and **Row**.

➤ **To read a value from file**

- 1.** Left click on the variable to be read in from the file.
- 2.** Click on the **Read value from a data file** button to open the file position setting part of the dialog.
- 3.** From within the **Data File Modules** box, select the data file that the value is to be read from.
- 4.** Place the cursor just before (and in the same row as) the required data item and highlight the item by dragging the mouse over it. Alternatively, type the position of the data item into the **Position of Variable in File** text boxes.

In Fig. 9-3, the variable “Latitude” is to be read from the file associated with the **DataFile** module **Meteorological data**.



**QueryFile** modules are restricted to reading information from a fixed (absolute) position in a file. In comma-separated data, such a fixed position for an item of information is not guaranteed across multiple files, and hence it should not be used for **QueryFile** data.

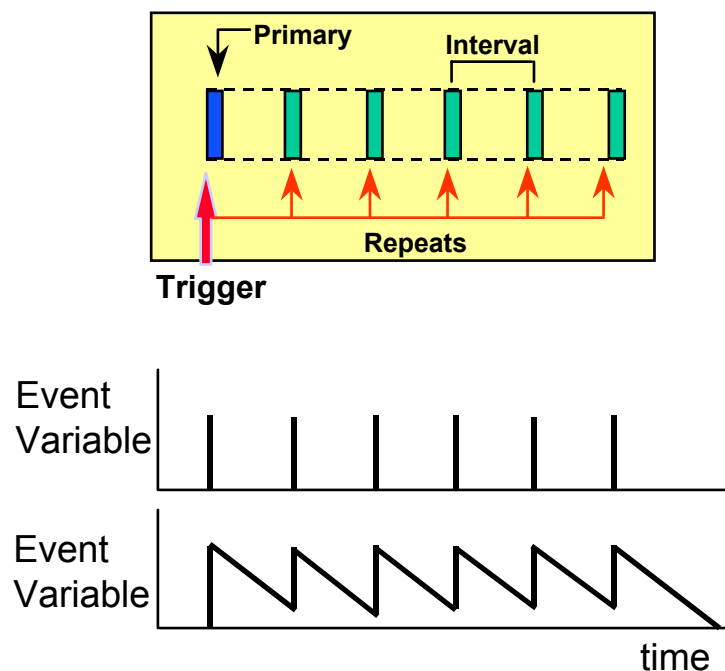
## 10 The Event module



The **Event** module allows events (such as management actions) to be performed at particular times during the simulation. For example, it may be used to schedule and perform pesticide spraying or harvesting of a crop. The timing or threshold conditions that trigger the event must be set before a simulation run (see *Initialising the Event module*, page 62). The event module can be set up so that events can be triggered differently for different subpopulations.

An event is triggered when the appropriate date or threshold is reached during a simulation run. Normally, the trigger action will initiate the event response immediately, but an **EventWithDelay** module will delay the initiation of the response by a period that is controlled by a factor (parameter, function or process). The **response factor** associated with the Event module, and the **Event Duration Override** setting (determined by the model designer), control the precise way in which the event is applied and runs its course after it has been triggered. The response factor is often a function or process to achieve effects such as an exponentially declining event effect.

**Fig. 10-1** An event incorporating several repetitions and its effect on the Event Variable



In Fig. 10-1, a single *Event* incorporating a primary event *action* and several repetitions is illustrated within the shaded box. It is initiated by a *Trigger* condition, indicated as a solid arrow. The trigger is the condition that initiates the event, for example a specified date or threshold condition. Each trigger can have a specified number of repetitions at a given interval associated with it. For example, consider the case of a chemical spray, where the user sets up a treatment (using a single Trigger) to occur yearly on March 1, repeated three

times at 14 day intervals. In this case all of the four sprays in that set are considered to belong to a single event, with the event scheduled to occur once each year. If the simulation is run for five years, a total of five Events will be triggered, which will translate to a total of 20 spray applications (actions). The two graphs in the lower half of Fig. 10-1 show the value of the output variable, **Event Variable**, under two scenarios. In the upper graph, a parameter, along with an **Event Duration Override** setting of one timestep, is used to specify the event effect, so that the output takes on the value specified by the parameter at the event action timesteps, but returns to zero immediately thereafter. In the lower graph, a function is used to determine the event effect, with characteristics chosen so that the output "decays" from the maximum value reached at each event action timestep.

Triggers can also be set to terminate the event action (i.e., return the Event Variable to 0). These **Off** triggers are specified in the same way as the normal **On** triggers described in the previous paragraph, either using dates or threshold conditions.

The **Event2**, **Event3**, **Event4** and **Event5** modules are exactly the same in their action as a normal **Event** module, but have more than one **Event Variable** (a number corresponding to their trailing digit) and an equivalent number of response factors. The event response is triggered in the same way as a simple **Event** module, but each output is controlled by its corresponding response factor. The **Event Duration Override**, if used, will set all the **Event Variable** outputs back to 0 at the same time, as will any occurrence of an **Off** trigger.

Input  
variables

The **Event** module has up to four possible input variables: **Day of Year**, **Simulation Date**, **Threshold**, and **Cost per instance**. The exact way in which timing of events is specified will depend on whether **Day of Year** or **Simulation Date** (or both) is used (see *Initialising the Event module*, page 62). If the Threshold input is used, events can be triggered by threshold conditions (e.g. we may want to spray a pest only when numbers exceed a predefined threshold). The **Cost per instance** input supplies the cost of each occurrence of an event (for example, the cost of one spray application). This defaults to one if the input is unused.

Event  
Response

What happens to the Event module's output variable (**Event Variable**) is determined by the corresponding **response factor**. This may be a parameter, or a function or process (response function). If a parameter is used to specify the response, the Event Variable will reflect the parameter's value on the event action timestep, and subsequent timestep, until reset to zero by the **Event Duration Override** setting (or an **Off** trigger). If a function or process is used instead, the value of the Event Variable at any timestep after an event action will be determined by evaluating the function or process used (once again terminated by the **Event Duration Override** setting or **Off** trigger, if applicable).

If a response function is used to determine the event outcome, a local variable (**Days since Event**) is available as a driving variable for that function, and may

have been used by the model designer. The variable represents the number of days that have elapsed since the event was last triggered.

Output  
variables

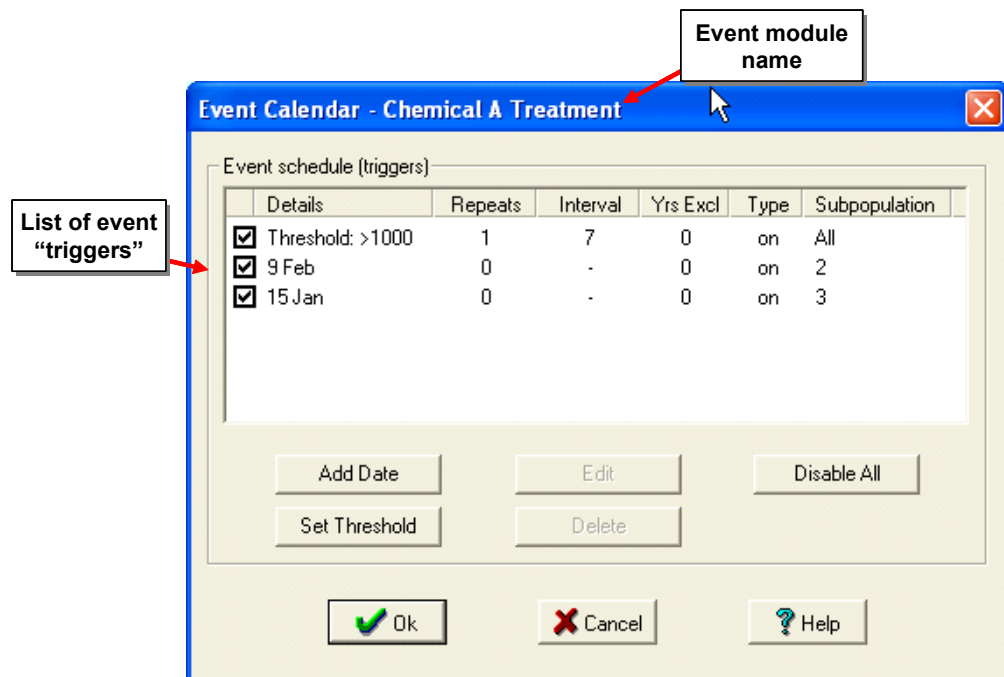
There are several output variables from an **Event** module. One is the **Event Variable**, which could be used to drive mortality in the lifecycle, for example. Note that the **Event** module has a single such output variable, while the **Event2**, ..., **Event5** modules have a number equivalent to the final digit of the module type. Another output variable is the **Event Cost**, which is used to calculate the “cost” of the event. Its value is equal to the input variable **Cost per instance** in those timesteps that an event action occurs, and is 0 at all other times. Finally, two more output variables acts as “flag” outputs, that indicate when an event has been triggered (see *Builder User’s Guide*).

## 10.1 Initialising the Event module



Initializing an **Event** module involves setting either the time that events are going to occur or specifying the threshold that triggers an event. To initialize the **Event** module left-click on its bitmap representation and select **Initialize Module** from the small menu that appears.

Fig. 10-2 An event initialization dialog box with two triggers specified but disabled.



The event initialization dialog box contains a window that lists the event triggers that are currently set.

The **Event schedule (triggers)** window contains the list of currently set event triggers. Each trigger item consists of details of the trigger condition (such as the date or threshold), the number of repeats required after the event is triggered and the interval between the repetitions (in days). It also shows whether the trigger is an **On** trigger (i.e., it initiates the event action) or an **Off**

trigger (i.e., it terminates the event action), as well as the number of years at the beginning of the simulation in which triggering of the event is to be suppressed (**Yrs Excl**). The latter is used in simulations where we want to trigger the events only after the population has settled down from the initial conditions. If the module takes part in the model's subpopulation structure, the name of the subpopulation affected by the trigger is also listed. A trigger may affect all subpopulations, in which case the word "All" is displayed.

Two buttons in the lower left of the dialog are available to add a new event trigger date or condition. Note that the **Set Threshold** button will only be present when the **Threshold** input variable of the **Event** module is used in a model. In other cases, where neither of the **Day of Year** or **Simulation Date** inputs are connected to the appropriate **Timer** variables, the **Add Date** button will not be present.

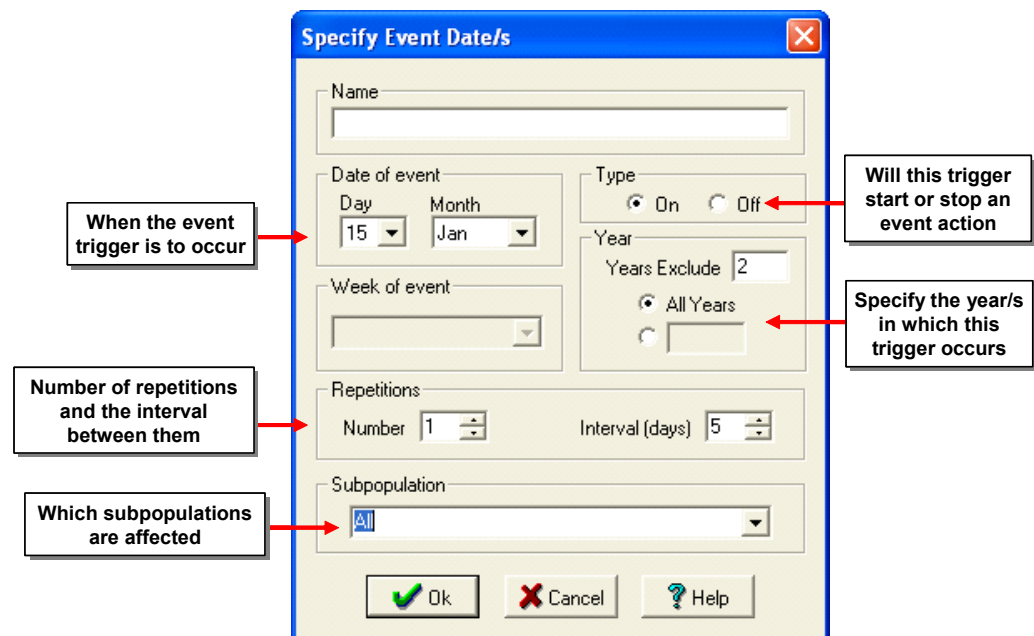
**Disable Event Triggers** The **Disable Event Triggers** check box allows the event triggering for the module to be disabled for subsequent runs, without losing the triggers that have been set.

### 10.1.1 Setting a Date triggered event

#### ➤ To add an event triggered on a particular date

1. Click on the **Add Date** button to obtain the **Specify Event Date/s** dialog (Fig. 10-3).

Fig. 10-3 An event date dialog box with example event with repetitions.



2. For models with a daily timestep, the **Day** and **Month** boxes will be enabled, while weekly timestep models will have the **Week of event** enabled. In the former case, select the **Day** and **Month** required from the

drop down boxes. In the latter, select the required week. The available weeks are shown within the drop-down box as the date of its first day, and a week number – for example, Jan 22 (4).

3. Select the type of trigger required on this date. An **On** trigger initiates an event action, an **Off** trigger terminates it.
4. Check one of the radio buttons within the **Year** box. Either the event occurs every year on the date specified in step 2 (**All Years**), or in one particular year, which must be typed into the text box. If the former is selected, a number of years at the start of a simulation run can be excluded by specifying this in the **Yrs Exclude** text box.
5. If the event action is to be repeated for the same trigger, indicate this by using the **Repetitions** area of the dialog to provide the **Number** of repetitions and the **Interval** between them (in days).
6. If the module uses sub-populations, select the subpopulation that this event trigger will be applied to. Select **All** if it is to refer to all sub populations.
7. Click **Ok** to return to the **Event Calendar** dialog. The new trigger will be listed in the Event Schedule.

Fig. 10-4 The resulting dates from the event set in Fig. 10-3.

JANUARY						
Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
			1	2 ✓	3	4
5	6	7 ✓	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

**Repetitions** Repetitions of events can be performed by increasing the **Number** box to a value greater than zero and setting the **Interval** to the number of days that the replicates are apart.

**All Years?** If **All Years** is selected then the event will occur each year that the simulation runs. If **All Years** is not selected, the event will only occur once – in the year specified in the text box.

Note that the trigger can be given a **Name**. However, this name is not currently used anywhere and may be safely left blank.

### 10.1.2 Setting a Threshold triggered event

Note that the current version of DYMEX allows only a single threshold trigger for each **Event** module.

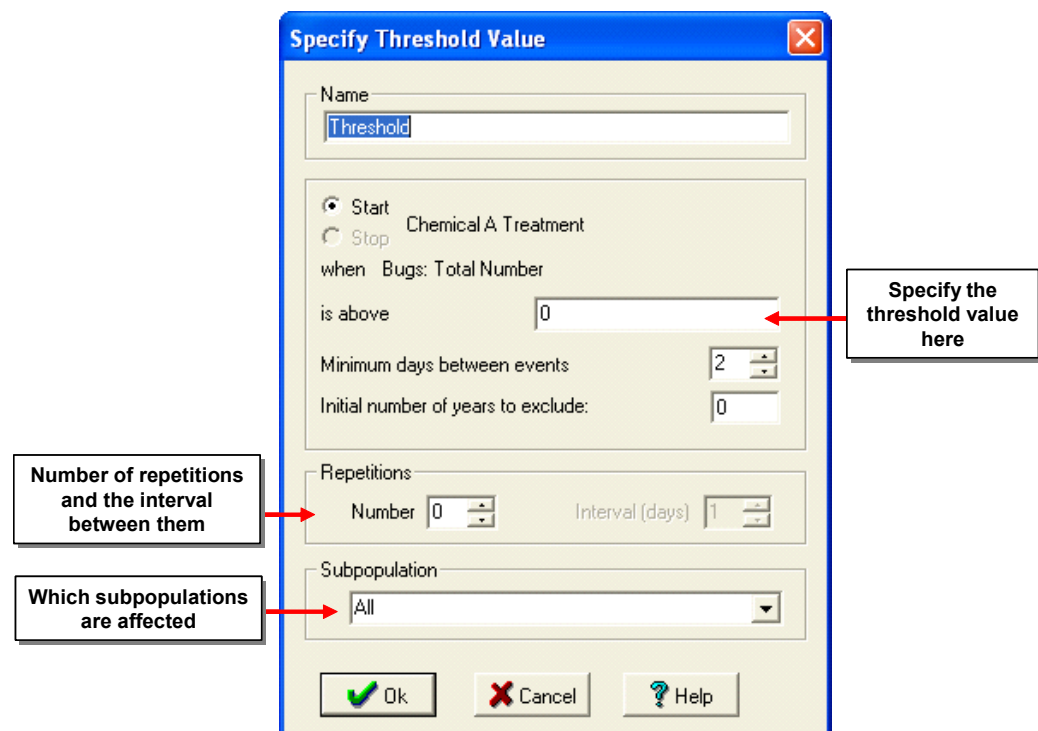
#### ➤ To add an event triggered by a threshold condition

1. Click on the **Set Threshold** button in the **Event Calendar** dialog.

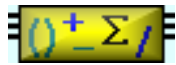
The resulting **Specify Threshold Value** dialog will vary slightly in appearance depending on the type of threshold condition that is set. The buttons at the top left (**Start** and **Stop**) indicate what effect the threshold trigger will have on the event (these correspond to **On** and **Off** conditions). Often, only one of these will be available, as specified by the model designer. The **Start/Stop** buttons are followed by the name of the Event module for which the condition is being specified. The 2<sup>nd</sup> line (**when**) nominates the variable that is being used to trigger the event. The 3<sup>rd</sup> line will be labelled either **is above**, **is below**, **crosses above** or **crosses below**, depending on the particular module and contains the text box into which the threshold value must be typed.

2. If a choice between **Start** and **Stop** is available, choose the required action for this trigger.
3. Type the value that should trigger an event into the text box. For example, if the text box is labelled **is above**, and an event is to be triggered when the threshold variable exceeds 100, that value needs to be entered.
4. Set the minimum time that may elapse between two consecutive events by typing the required value in the text box labelled **Minimum days between events**. Note that in weekly models, the number of weeks between events must be supplied (the label changes appropriately).
5. A number of years at the start of the simulation run can be excluded (even if the trigger condition is satisfied) by specifying this in the **Initial number of years to exclude** box.
6. If the event action is to be repeated for the same trigger, indicate this by using the **Repetitions** area of the dialog to provide the **Number** of repetitions and the **Interval** between them.

Figure 10-5 An event threshold dialog box with an example event containing repetitions.



## 11 The Expression module

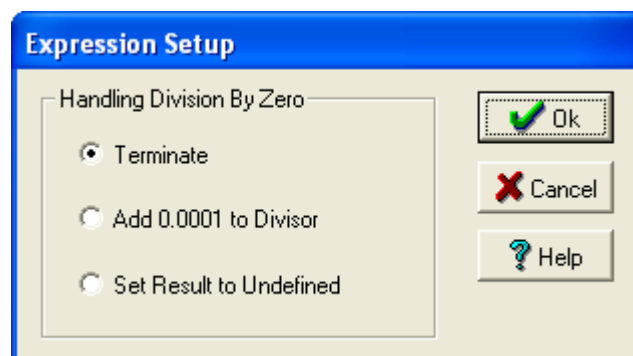


The **Expression** module performs simple mathematical operations on the input variables. The module can be used to manipulate variables with subpopulation structure. The exact operation being performed has been set by the designer of the model and cannot be changed by the model user. It can be examined by clicking on the module's bitmap representation, and selecting **Module Information** from the resulting popup menu. Note also, each of the input variables (x) listed in that dialog, and whether they are used directly as input, or are first negated (i.e., -x is used as input) or inverted (i.e., 1/x is used as input).

Most Expression modules require no initialization, and the **Initialize Module** option will be absent from the module's popup menu. However, if one or more of the inputs to the module are inverted before use in the expression, there is a possibility of a division by 0. How this condition is to be handled is specified in the **Builder** in Version 2 of DYMEX. Therefore it is only in models created using an earlier version of the **Builder** that any initialisation may be necessary.

The **Expression Setup** dialog (Fig. 11-1) provides 3 choices for handling this situation. The simulation can be directed to terminate, which is the default choice. That option would be appropriate in the case where the user knows that the input in question can never be 0 in a normal situation, so that the occurrence of that condition is treated as an error. The second option (**Add 0.0001 to Divisor**) should be used in the situation where we know that the output from the expression module is used as input to another module, but it is legitimate for the divisor to be zero (it may, for example, be the number of individuals in some lifestage). The last option (**Set Result to Undefined**) would be used for the same situation as the previous one, but where the output from the expression is not used as input to another module.

Fig. 11-1 The Expression setup dialog





## 12 The Function module

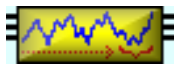


The **Function** module most commonly uses one of the functions provided in DYMEX (or a user-defined function) to perform an operation on an input variable (the driving variable of the function) . The module can be used to manipulate variables with subpopulation structure. There will be a number of parameters associated with the module, depending on the function being used. As usual in DYMEX, a simple parameter or a process may be used instead of the function (see Section 2.4). The values of any parameters associated with the module can be changed by clicking on the module bitmap and selecting **Show Parameters** from the popup menu (see *Modifying Parameters*, page 76). The Function module requires no other initialization before use in a simulation run.

The **Function2**, **Function3**, **Function4** and **Function5** modules are exactly the same in their action as a normal **Function** module, but have more than one output variable (a number corresponding to their trailing digit) and an equivalent number of factors. Each output is controlled by its corresponding factor. These modules in effect are the same as the equivalent number of simple **Function** modules.

Function modules may take part in the model's subpopulation structure.

## 13 The Running Mean module



The **Running Mean** module calculates a running mean or running total over a specified period of time. There are no settings that the user can adjust in the *Simulator*. However, the **Module Information** dialog contains information on whether a mean or total is being calculated, and the period over which the calculation is being made.

## 14 The DegreeDay module



The **DegreeDay** module is a simple module with no inputs and one output variable. The output variable reflects the number of degree-days accumulated above a threshold temperature, given by the module's parameter. The threshold value can be changed by clicking on the module bitmap and selecting **Show Parameters** from the popup menu (see *Modifying Parameters*, page 76). The Function module requires no other initialization before use in a simulation run.

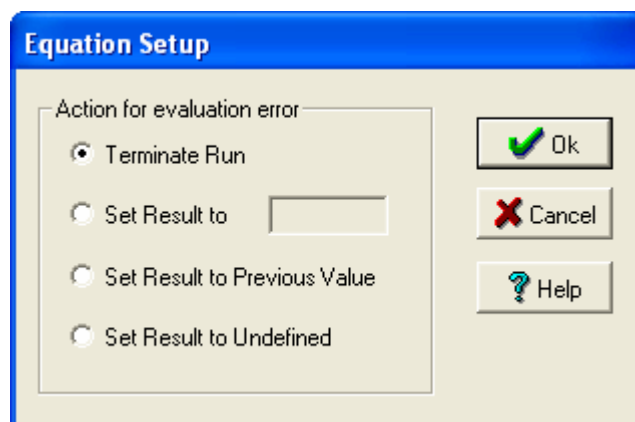
## 15 The Equation and Counter modules



The **Equation** module evaluates an equation defined in the DYMEX equation syntax. The module can be used to manipulate variables with subpopulation structure. There will usually be a number of parameters associated with the module, depending on the equation being used. The values of the parameters can be changed by clicking on the module bitmap and selecting **Show Parameters** from the popup menu (see *Modifying Parameters*, page 76). For models built using Version 2 (or later) of the **Builder**, no other initialisations are required or available. For models built using earlier versions of the **Builder**, initialisation consists of specifying how error conditions during equation evaluation are to be handled. Such errors include a division by zero, calculating the logarithm of a negative number, etc.

The **Equation Setup** dialog (Fig. 15-1) provides 4 choices for handling an error in evaluating the equation. The simulation can be directed to terminate, which is the default choice. That option would be appropriate in the case where the user knows that the equation should evaluate correctly in a normal situation, so that the occurrence of that condition is treated as an error. The second option (**Set Result to**) should be used in situations where it is known the equation evaluation may fail, but should produce a known result in that case. An example would be if we were modelling resistant and non-resistant strains of a species, and the equation was used to calculate the proportion of resistant individuals in the population. When the total population is zero, the equation evaluation would fail, but a value of 0 may be preferable in that case, especially if that result is used as input by another module. The third option (**Set Result to Previous Value**) sets the result of any evaluation error to the output from the module at the previous time step (or 0 if it is the first time step). The final option (**Set Result to Undefined**) could be used for situations where an invalid result is expected and the output from the module is not used as input to another module.

Fig. 15-1 The Equation setup dialog



The **Counter** module is very similar to the **Equation** module, in that it evaluates an equation. However, the result of the equation evaluation is not used directly as output, but as a condition for incrementing a counter. During

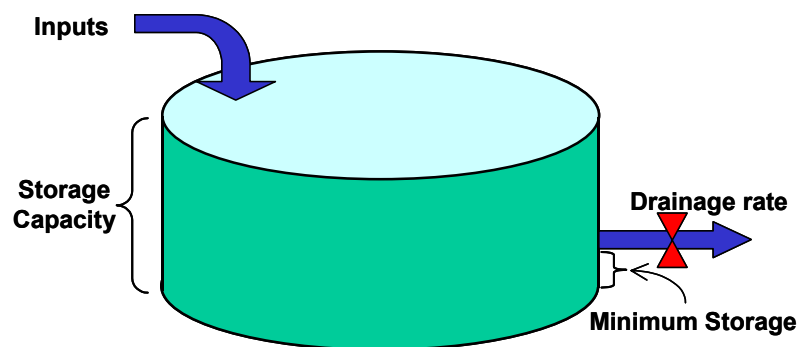
every time step that the equation evaluates to a positive value, the counter is incremented by a specified value. During all other time steps, the counter's value remains unchanged. The counter is always reset to 0 at the start of a simulation. The **Counter** module's output at any time reflects the current value of the counter. As with the **Equation** module, here will usually be a number of parameters that can be adjusted. These can be changed by clicking on the module bitmap and selecting **Show Parameters** from the popup menu. No other initialisation is required for this module.

## 16 The Storage module



The **Storage Container** module simulates a storage device with a nominated capacity (in specified units, eg  $\text{m}^3$ ), which is filled (or emptied) using rates determined by the input variables and a "Drainage rate" factor. If it is filled above capacity, the excess overflows, while a minimum storage capacity can also be set, beyond which the container cannot be emptied. Fig. 16-1 illustrates this module schematically.

Fig. 16-1 Schematic diagram of the operation of the Storage Module.



Up to four input variables each provide a "Fill Rate", the amount (in the same units as the **Total Storage Capacity**) being added to the store during a timestep.

The module has one or two outputs. The first (**Current Storage**) is the current amount in the store, while the second (**Overflow**) is the amount lost during the current timestep due to the **Total Storage Capacity** being exceeded. These outputs can be in the same units as the Storage Capacity, or scaled as proportions of Storage Capacity, as determined by the model designer.

Three factors have to be set by the user. As usual, the factors can be parameters, functions or processes.

- **Total Storage Capacity** (permitted range:  $>0$ ): The total capacity of the store.

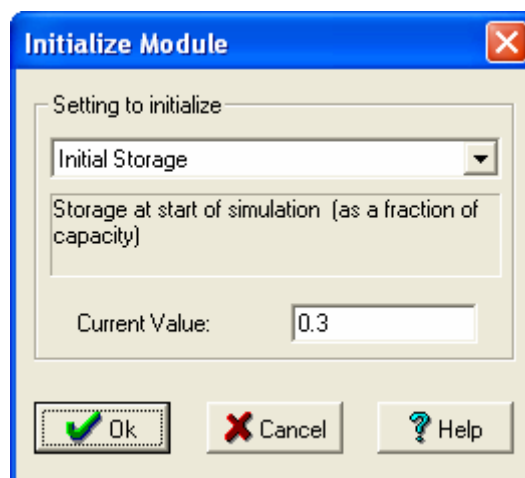
- **Minimum Storage** (permitted range: 0-**Total Storage Capacity**): The minimum level that the store may drop to if filled above that level. In other words, no drainage can occur until this minimum level is reached. The units for this factor are the same as those for **Total Storage Capacity**.
- **Drainage Rate**: This factor determines the amount of store contents lost per timestep. If the module uses “Direct” drainage, then the factor is in the same units as the **Total Storage Capacity**. Otherwise, the amount drained from the Storage is evaluated as the difference between the current Storage and the **Minimum Storage**, multiplied by the value of the factor.

## 16.1 Initialising the Storage module

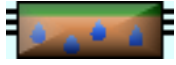


The **Storage** module is simple to initialize. The user can set the initial store (i.e., the amount in the store at the start of a simulation), which has a default value of 0.5 (being 50% of the **Total Storage Capacity**). Initialization is achieved by clicking on the module bitmap, selecting Initialize Module from the popup menu, and typing the required value into the Current Value box (Fig. 16-2).

Fig. 16-2 The Storage initialization dialog.



## 17 The Soil Moisture module



The **Soil Moisture** module is a sub-model that simulates the water balance in a single layer of soil. The module can be used to manipulate variables with subpopulation structure. It outputs an index between 0 and 1, with 0 being dry soil and 1 being a totally saturated soil (i.e., the output is available soil moisture as a proportion of soil moisture holding capacity). The input variables are total rainfall and evaporation for the timestep (both in mm). Before the module is used in a simulation run, it can be initialized with an initial moisture content and its parameters can be adjusted. Clicking on the module bitmap and selecting **Initialize module** or **Show Parameters**, respectively, from the popup menu is used to do this.

The three parameters that can be adjusted by the user are:

- **Soil Moisture Capacity** (permitted range: 50-200 mm): The water holding capacity of the defined layer of soil (say, 1 metre or to required rooting depth), in mm. Clay soils will have larger values than sandy soils. Sandy soils may store only 50 mm while loams may hold 150 mm and clays 200 mm within the rooting depth of most crops.
- **Evapotranspiration Coefficient** (permitted range: 0.5-1.2): Sum of soil evaporation and transpiration from vegetation expressed as a proportion of open Pan Evaporation. Generally between 0.5 and 1.2 with 0.8 typical.
- **Basal Evaporation** (permitted range: 0-5 mm): The portion of the evaporation input that is available for evapotranspiration regardless of soil moisture status (in mm). Values above 0 will empty water out of a non-saturated soil at higher rates than if it is set to the default of 0. Generally set to 0.

It is as well to note that in some models, one or more of the above parameters may have been replaced by a function. For example, a variable describing the biomass of vegetation may be used in a particular model, and the Evapotranspiration Coefficient may be driven using this variable with a functional relationship. This will become evident in the Parameter Window, where any such functional relationship will be illustrated and documented, and additional parameters may allow the function to be adjusted.

See the *DYMEX Builder User's Guide* for a detailed explanation of the workings of this module.

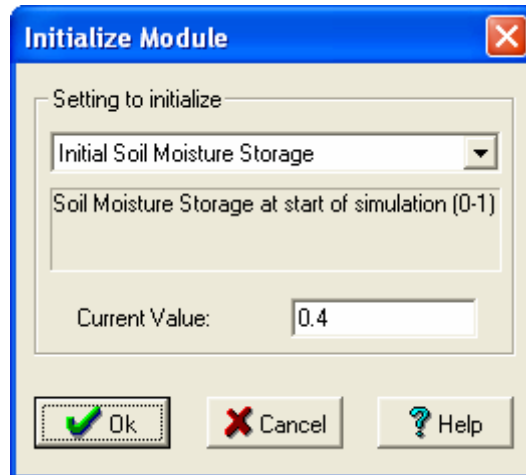
### 17.1 Initialising the Soil Moisture module



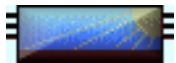
The **Soil Moisture** module is simple to initialize. The user can set the initial soil moisture, which has a default value of 0.5 (being 50% of soil moisture capacity). Note that for most circumstances, the default value is adequate, as the initial value will affect the soil moisture output for this module for only a relatively few timesteps. Initialization may be required if the model is being

run for a very short time, or it is required to simulate a known set of soil moisture data. Initialization is achieved by clicking on the module bitmap, selecting Initialize Module from the popup menu, and typing the required value into the Current Value box (Fig. 17-1).

Fig. 17-1 Soil Moisture initialization dialog box.

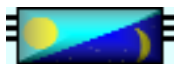


## 18 The Evaporation module



The **Evaporation** module calculates the class A pan evaporation (in mm/timestep) using the formula from Fitzpatrick, E.A. (1963), J. appl. Meteorol., 780 as adapted by Sands, P. and Hughes, R.D. (1976), Agricultural Meteorology, 161. The input variables used by this module are minimum and maximum daily temperatures (both in degrees Celsius), 9am relative humidity, 3pm relative humidity, and daylength (in hours). No initialization is required or available for this module, nor are there any user-adjustable parameters.

## 19 The Daylength module



The **Daylength** module inputs the latitude and either the day of year or "Simulation Date" and calculates the number of hours between sunrise and sunset. An optional second output provides the change in daylength since the previous timestep. The module has no user-adjustable parameters and requires no initialization. *Note that if "Simulation Date" is not provided as input, the day length change output will contain small discontinuities at the boundaries between normal and leap years.*

## 20 The Climate Change Scenario module



The **Climate Change Scenario** module takes as its inputs variables that represent current weather conditions (minimum temperature, maximum temperature, rainfall and evaporation) and outputs corresponding variables for weather conditions given a particular *climate change scenario*. The scenario is provided as a set of parameter values. The values of the parameters associated with the module can be changed by clicking on the module bitmap and selecting **Show Parameters** from the popup menu (see *Modifying Parameters*, page 76). The module requires no other initialization before use in a simulation run. The *Builder User's Guide* and DYMECH Help System contain a detailed description of the meaning of each parameters, as well as the algorithms used in calculating the outputs from the inputs.

For calculating the climate change scenarios, the module uses an *Equatorial Zone (or region)*. Different climate change effects can be set for locations inside and outside the zone. The extent of the Equatorial Zone is set in the **Model Options** dialog (see Section 25.3). Other modules may also use this setting.

## 21 The DemeSplitter module



The **DemeSplitter** module can only be used in models that use more than one sub-population. Its function is to break a variable with sub-population structure (a *demed* variable) into its component parts. Each output variable is a normal single-valued variable that can then be used as input to other modules that do not take part in the model's sub-population structure. The module does not have any parameters and does not require initialization before a simulation run. Fig. 21-1 shows a schematic representation of an instance of this module in a model with 3 sub-populations. The single input variable ( $V$ ) is a *demed* variable with components each representing one of the sub-populations. The value of the first output variable  $V_1$  will contain the value of the first sub-population component of  $V$ , and so forth.

Fig. 21-1 Diagrammatic representation of a DemeSplitter module in a model with 3 sub-populations.

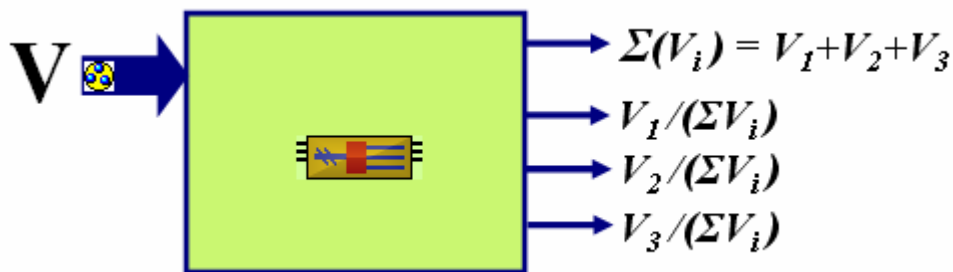


## 22 The DemeStatistics module



The **DemeStatistics** module (like the **DemeSplitter** module) can only be used in models that use more than one sub-population. If the model contains  $n$  sub-populations, then the DemeStatistics model has  $(n+1)$  output variables. Each of these output variables is a single-population (non-demed) variable. The first output variable is the sum of the values of each component of the input variable ( $V$ ),  $\Sigma V_i$  (see Fig. 22-1). Each of the other output variables is the proportion of the total represented by that component of the input variable (for example, the 3<sup>rd</sup> output variable is the proportion of the sum of all component values represented by the second component of the input variable (Fig. 22-1). The DemeStatistics module has no parameters and does not need initialization before a simulation run.

Fig. 22-1 Diagrammatic representation of a DemeStatistics module in a model with 3 sub-populations.



## 23 The SummaryManager module



A **SummaryManager** module is present in all models. It is responsible for updating the value of those Summary Variables in the model that summarize the value of other output variables directly (refer to Section 2.2 and Fig. 2-2). These Summary Variables are created in the **Builder** (see *Builder User's Guide*, Section 11) and cannot be modified in the **Simulator**. The SummaryManager module requires no initialisation. To see what Summary Variables have been defined for the model, click on the SummaryManager bitmap and select **Show Summary Variables** from the resulting menu. This shows the **Summary Variables** dialog (Fig. 23-1).

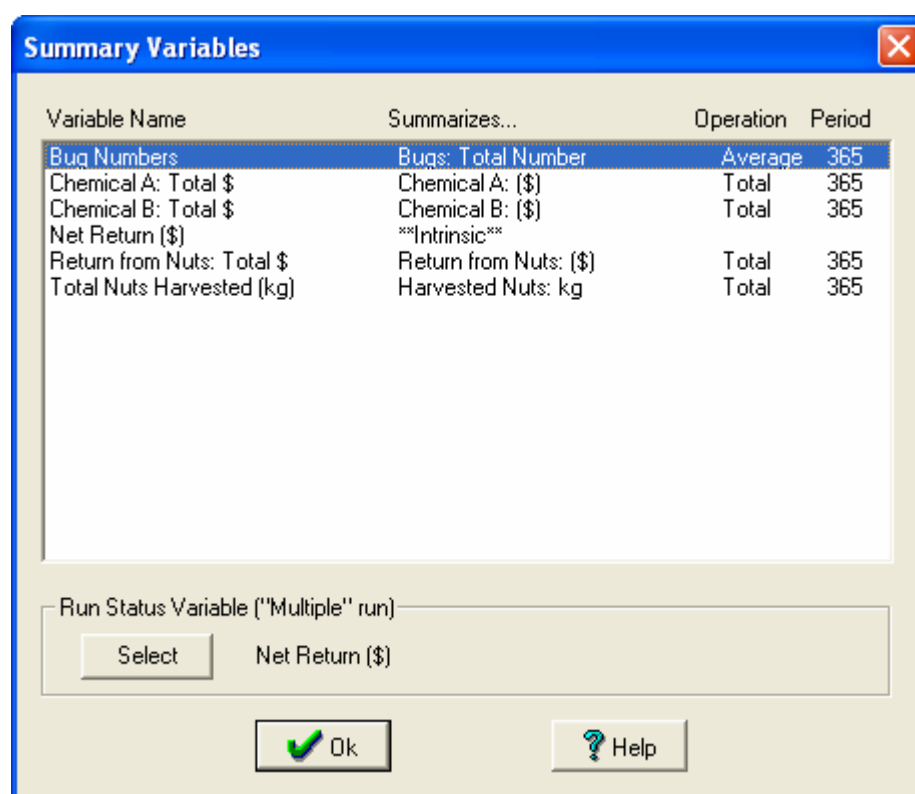
The dialog lists all of the model's Summary Variables in the large list box at the top. For each Summary Variable, the variable being summarized is shown in the **Summarizes** column, and the summary operation performed and summary period are shown in the **Operation** and **Period** columns, respectively. The **Period** is always given as  $n$  days, corresponding to the last  $n$  days of the simulation. The **Operation** shown will be one of the following:

- **Average** - is average of the variable during the period shown.
- **Last Value** - is the last value the variable attains during the simulation. Note that the period is irrelevant for this statistic.



- **Max** - is the maximum value of the variable during the period shown.
- **Min** - is the minimum value of the variable during the period shown.
- **Total** - is the total of the variable during the period shown.
- **\*\*Intrinsic\*\*** - indicates that the variable is not a direct summary of a model variable but the output from a Summary module. Summary modules are listed after the SummaryManager module in the Component Window (and have the same light-green background colour as that module).

**Fig. 23-1 A Summary Variables dialog box with two selected variables.**



During execution of a **Multiple** run, it is possible to display the value of one Summary Variable on the **Run Status** dialog (the dialog that shows the progress of the run). The variable is selected from the Summary Variables dialog.

➤ **To select a variable for display in the Run Status dialog**

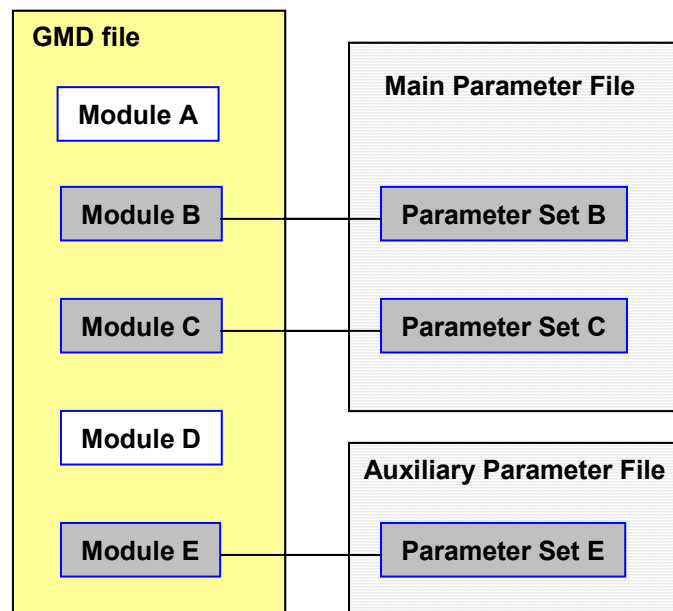
- 1.** Select it from the list by clicking on it
- 2.** Click on the **Select** button. Its name will appear beside the button.

## 24 Modifying Parameters

Parameters are constants that allow a model to be adjusted for particular situations. In general, the model designer will set a default value of each parameter that represents his or her best estimate for that parameter, and a range that indicates to some extent the degree of uncertainty for that parameter. Another reason for setting a range may be so that the model (which was designed for one particular species) can be adjusted to suit another species with the same biological attributes.


Each parameter belongs to a module, and a set of all the parameters needed by a module is termed a *Parameter Set*. The Model Description File (GMD file) contains the context, default value and range of all the parameters used in the model. These default values constitute the *Default Parameter Set* for each module. In general, the actual values of the parameters for any particular simulation run are stored in one or more *Parameter Files* (usually with the extension, .gmp). One *Main Parameter File* (usually with the same name as the Simulation File, but extension .gmp) is always present, and most modules store their parameter values in this file. However, some modules may store their values in an *Auxiliary Parameter File*. See Section 2.6 for a description of these files. When parameters are modified in the **Simulator**, and the changes are saved, the values are modified only within the associated Parameter File – the GMD file is not changed.


**Fig. 24-1 The storage of parameters for a hypothetical model consisting of 5 modules, 3 of which have parameters.**



The status of each module's parameters are indicated using a "Parameter icon" in the Model Components window, placed just to the left of the Module bitmap. If the parameter icon is absent, the module has no parameters.

Otherwise, the type of icon shown indicates the parameter status, as indicated below:

 The module currently does not have any valid parameter settings. This condition can only occur if an **Auxiliary Parameter File** has been specified for the module and the **Default Parameter Set** has been declared invalid. Note that unless the module is optional, the model will not be able to run unless a valid parameter set is loaded.

 The module is using the **Default Parameter Set** to determine its parameter values. Note that this means that the user will not be able to change the value of any of the module's parameters.

 The module is using the **Main Parameter File** for storing its set of parameters.

 The module is using an **Auxiliary Parameter File** for storing its set of parameters.

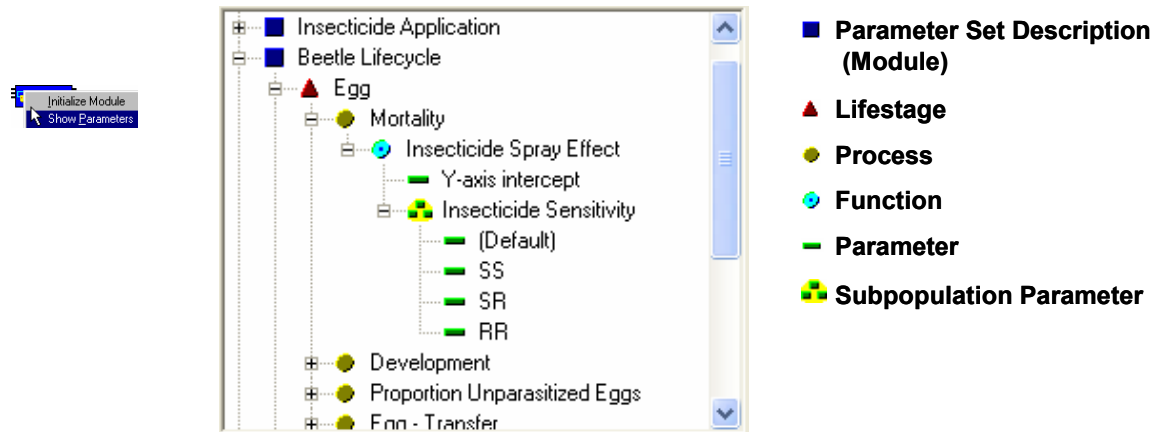


All parameter files are created and maintained by the **Simulator** program – the **Builder** is completely unaware of them. This is the cause of a commonly encountered problem, where the model designer changes one or more parameter default values in the **Builder**, saves the model, loads it in the **Simulator** and finds the parameter values have not changed. In actual fact, the default values have changed, but the **Simulator** has overridden them with the values stored in the Parameter File (which were most likely the previous defaults). The simplest way to avoid this problem is to check the **Use Parameter Defaults (not parameter file)** option in the **Model Preferences** dialog (see Section 25.3), which will cause the **Simulator** to ignore the parameter file completely. The disadvantage of this method is that the option must then be unselected before the **Simulator** can be used to adjust parameter values. Alternatively, the parameter file could be deleted after changes are made to the parameter defaults in the **Builder**.

## **24.1 The Parameter “Tree” Window**

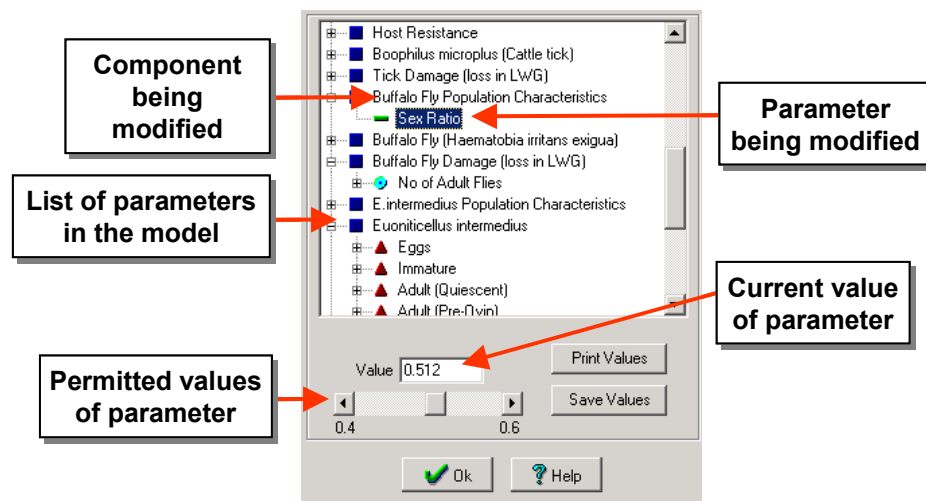
Parameters in DYMEX are modified from the **Parameter “Tree” Window**. The window is accessed in one of three ways: (1) by clicking on the “Parameter icon” of the appropriate module, (2) from the local menu of any module that contains parameters by clicking on the module bitmap and picking **Show Parameters** from the resulting popup menu, or (3) from the **Initialization** menu, by picking the **Model Parameters** option. Each of these actions results in a Parameter Window that displays all of the model's parameters, but the first two methods will select and highlight the first parameter of the selected module, thus removing the need to find it in a long list of parameters.

**Fig. 24-2 A parameter list, showing the Egg Mortality process. The symbols corresponding to the different node types are also shown.**



The caption of the Parameter “Tree” Window includes the name of the *Main Parameter File* being used. The parameters in the Parameter Window are listed at the left, in a “tree” view, where various model components form the nodes of the tree, with the parameters forming the terminal nodes. This allows the “context” of a parameter to be determined (as several parameters may share the same name). Nodes can be “opened” in this view by clicking on the small rectangle containing a + in front of the node symbol. Each node type (eg, module, process, etc.) is indicated by a symbol in front of the nodes name. These are listed in Fig. 24-2. Any node can be selected by clicking on it with the mouse. The currently selected node’s name is then highlighted.

**Fig. 24-3 A Parameter list box within the Model Parameters dialog box.**



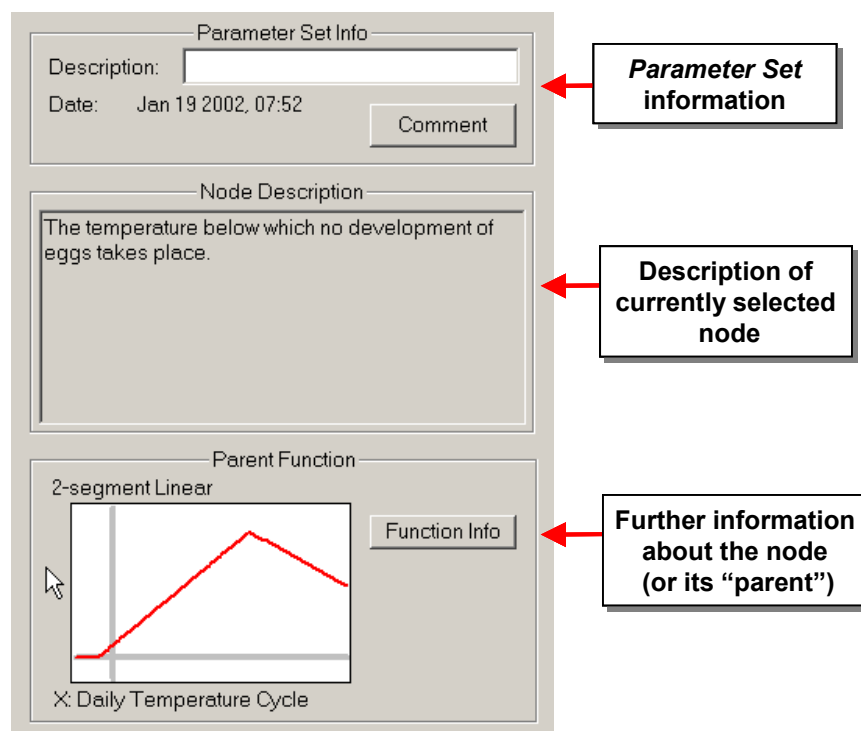
The right of the dialog contains several panels with information regarding the currently selected node. This information varies depending on the type of node that is currently selected. A detailed description of these panels is given later in this section.

➤ **To modify the value of a parameter**

- 1.** Open the Parameter Window by one of the methods described in the previous paragraph.
- 2.** Select the required parameter node from the **Parameter List** box (Fig. 24-3). You will need to “open” the parameter’s containing nodes to get to it.
- 3.** Change the parameter’s value using either the **Value** text box or the scroll bar.

When a parameter is selected, its value is displayed in the **Value** box and its valid range will be given at either end of the slider control. If the slider control is disabled, then either the parameter is fixed (its value cannot be changed, and the **Value** box will be disabled also), or the model designer has not provided a lower or an upper limit to the range of allowed values (and its value can be changed only by typing directly into the **Value** text box). Note that the **Value** box and slider are disabled if a node other than a parameter node is currently selected.

**Fig. 24-4** The information section of the Model Parameters dialog box.



The panels in the right-hand half of the dialog display various items of information relating to the currently selected node (Fig. 24-4). The top panel (**Parameter Set Info**) provides information about the *Parameter Set* that the currently selected node belongs to. Each *Parameter Set* has a Description, Date, Comment, and may be “Protected”. If no description is provided the **Description** edit box is left blank. Note that the top-level nodes in the “tree” view (which correspond to modules) show the module name if no *Parameter Set* description is supplied – otherwise they show that description. A

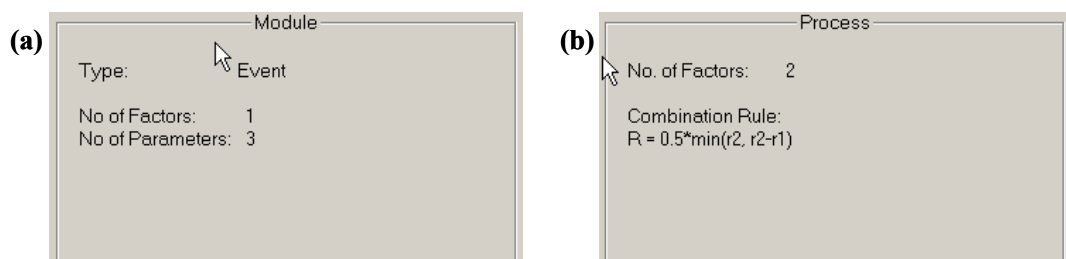
description can be useful if the parameters in the set describe a particular module configuration. For example, in a Soil Moisture module, the parameters being used might correspond to sandy soil. In that case, the parameter set could be described as such (for example “Soil Moisture – sandy soil”). The **Comment** button opens a dialog that can be used to provide or edit a longer description for the Parameter Set. The comment is in a “rich-text” format and can contain formatting features such as different fonts. The **Date** is updated automatically each time that the parameter file is saved. If the word **PROTECTED** appears in the panel (under the **Date** string), the Parameter Set cannot be modified in any way from within DYMEEX.

The centre panel (**Node Description**) provides the description for the currently selected node as set in the GMD file. If the model designer has provided no such description, the text “No description available” is shown.

The lower panel is labelled in various ways depending on the node type selected, and provides extra information about the associated node or its parent node, as follows:

- (1) If a module node is selected, the panel will be labelled **Module**, and lists the module type, as well as the total number of factors and parameters contained in it (Fig. 24-5a).
- (2) If a process mode is selected, the panel is labelled **Process** and lists the number of factors that constitute the process and the *Combination Rule* used to combine the factors (Fig. 24-5b).
- (3) If a function node is selected, the panel is labelled **Function** and shows a diagram of the shape of the function (Fig. 24-4). The small button on the right (**Function Info**) can be used to obtain more information about the function shape (such as, for example, the equation describing the function). The driving variable for the function is given below the function graphic.
- (4) If a parameter node is selected, the panel is labelled **Parent Function** or **Parent Process**, depending on whether the parent node is a function or process. In either case, the panel contains information on the parent node.

**Fig. 24-5 The node information panel for (a) a module of type Event, (b) a process composed of 2 factors.**



- The **Parameter List** lists all the parameters used within the model. Even though there is some contextual information that identifies the parameter's position in the model, it can be seen how important it is to give good names and adequate descriptions to parameters and functions when constructing a model using the *Builder*.
- By left clicking on the **Print Values** button (Fig. 24-3) or choosing **Print** from the **File** menu while the Parameter window is active, all the parameters and their values can be printed. The parameters are grouped under their module name (and lifestage and process for Lifecycle module parameters), and the names of driving variables and function shapes are provided for function parameters.
- When parameter values are modified, the values in the Parameter (GMP) file are not automatically changed to reflect these new values. This can be done by left clicking on the **Save Values** button, or selecting **Save** from the **File** menu while the parameter window is active. The parameters can also be saved to another file by selecting **Save as** from the **File** menu. If parameter values are changed but not saved, the user will be reminded that the parameter values have changed when exiting DYMEX or closing the model, and given the opportunity to save them then. Note that if a model using an Auxiliary Parameter File is being used, then **Save** operations will update only the files actually containing changed parameter values.
- Any parameter file that is compatible with the currently loaded model can be opened using the **File** menu's **Open** command when the Parameter window is active. This changes all the parameter values in the model to those in the new file, making the new file the active Parameter file. This can be useful in cases where alternative sets of parameter values are saved in different files, either for related species or as backups when trying different values during model optimization.
- The most recently saved parameter values can be restored at any time by making the Parameter window the active window and selecting the **Revert to Saved** option from the **File** menu.

## 24.2 The Parameter "Grid"

Parameters can also be viewed and modified from an alternative, "grid"-type window. This may be more convenient for modules that do not have a complex hierarchical structure of processes and functions. The parameter grid view is enabled by checking the "**Use Parameter Grid for non-Lifecycle modules**" option in the **Current Model** preferences (see Section 25.3). When this is done, clicking on a module's parameter icon or picking **Show Parameters** from the popup menu that results from by clicking on the module bitmap will lead to the Parameter "Grid" window (Fig. 24-6). Note that (as indicated in the option) the Parameter "Grid" window cannot be used to display Lifecycle module parameters, nor can it be used to display parameters of modules that use subpopulations.

The view shows all parameters, grouped under the module factors (the blue cells, for example “Soil Moisture Capacity”). The parameter names are listed in abbreviated form. The full name of the parameter can be obtained by moving the mouse cursor over the name field and leaving it there for about a second. This will cause a small window with the full name to appear next to the cursor (Fig. 24-6).

Parameters are modified by double-clicking on the parameter value field and typing the new value. Typing within a field should be terminated with either the “Enter” or “Tab” key (or clicking on another grid cell). The window is closed using the “Close” button at the top right.

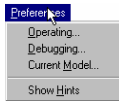
**Fig. 24-6 The Parameter “Grid” window for a 1-layer Soil Moisture module.**



The parameters can be saved (if they have been adjusted) or printed from the **File** menu.



## 25 Simulator Preferences



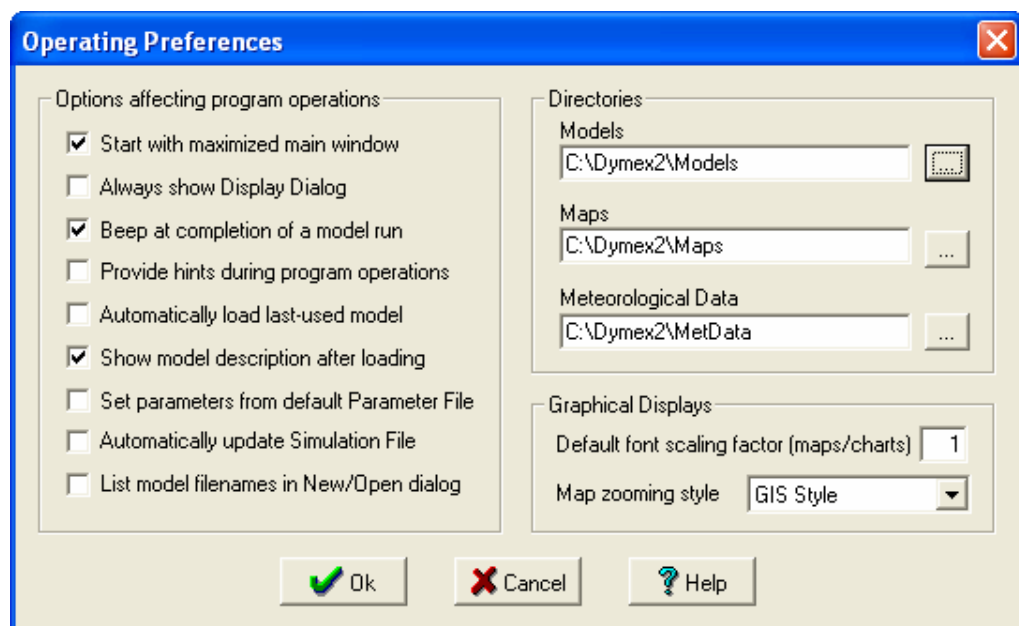
The **Simulator Preferences** menu provides settings and options that allow the user to optimize the operation of the **Simulator** to suit their own computing environment and situation. The available options are grouped into three major categories, available from the **Operating**, **Tracing** and **Current Model** sub-options, respectively. Each of these is dealt with in separate sections below.

### 25.1 Operating Preferences

Choices in the **Operating** preferences affect both program operation and the model window appearance. Some preferences are set as defaults when the **Simulator** is started for the first time. The current settings of these options are retained in Windows Registry.


The left panel in the Operating Preferences dialog deals with options that determine some operating characteristics and the appearance of the main **Simulator** window (Fig. 25-1).

Fig. 25-1 The Operating Preferences dialog box.



- **Start with maximized main window** - when the **Simulator** is started it will be in the maximized state, filling the whole screen.
- **Always show Display Dialog** - show the display dialog (i.e., table, chart, map or file edit dialogs) before that display is created even when the display format is a previously created and saved format. If this option is not checked and a saved format is selected, the selected display is shown immediately.

- **Beep at completion of model run** - when a simulation has been completed the program will make a beep sound.
- **Provide hints during program operation** - when the *Simulator* is running the hints window will provide useful hints on how to use the program.
- **Automatically load last-used model** - when the *Simulator* is started, the last model used in the previous session will be load automatically. This is particularly convenient for users that use only a single model.
- **Show model description after loading** - shows the Model Description dialog box immediately after a model is loaded, and before the Component Window is opened.
- **Automatically update Simulation File** – the Simulation File will automatically be updated on disk any time a user setting is changed or a format is saved.
- **List model filenames in New/Open dialogs** - the New and Open Simulation dialogs, rather than showing the names of the models in the *Models* list at the left, show the names of the Model Description (*gmd*) Files.

The **Directories** panel shows the names of the directories that DYMEX is currently using when certain operations are performed. To change any of these directories, the new directory name can be typed into the corresponding edit field, or the **Browse** button () to the right of each edit field may be used to navigate to the required directory.

- **Models** - the directory where DYMEX expects to find its models and Simulation files. The **New** and **Open** operations look for model files in this directory and any of its subdirectories.
- **Maps** - the location of map definition files and shape files for maps.
- **Meteorological data** - the location of meteorological data files, including *MetManager* data files.

The **Graphical Displays** panel provides options that affect the graphical displays in the model (charts, maps and lifecycle diagrams).

- **Default font scaling factor (maps/charts)** - supplies a scaling factor that is applied to the font size of all graphical displays. For example, to make all text in these displays larger by 50%, set this factor to 1.5.
- **Map zooming Style** – the setting of this option determines how zooming operates in maps. With **GIS Style**, the clicked point (or centre of zooming rectangle) becomes the centre of window in the zoomed map, while in the **Dymex Styles** the clicked point (or centre of zooming rectangle) after the zoom is at the same location on the screen as before the zoom. The zoom rectangle has the same aspect ratio as the map in **Dymex Style 2**, while its

aspect ratio is set by the mouse position for the other styles. Note that zooming using the mouse wheel is only available if either of the **Dymex Styles** is set.

## 25.2 Model Trace Options

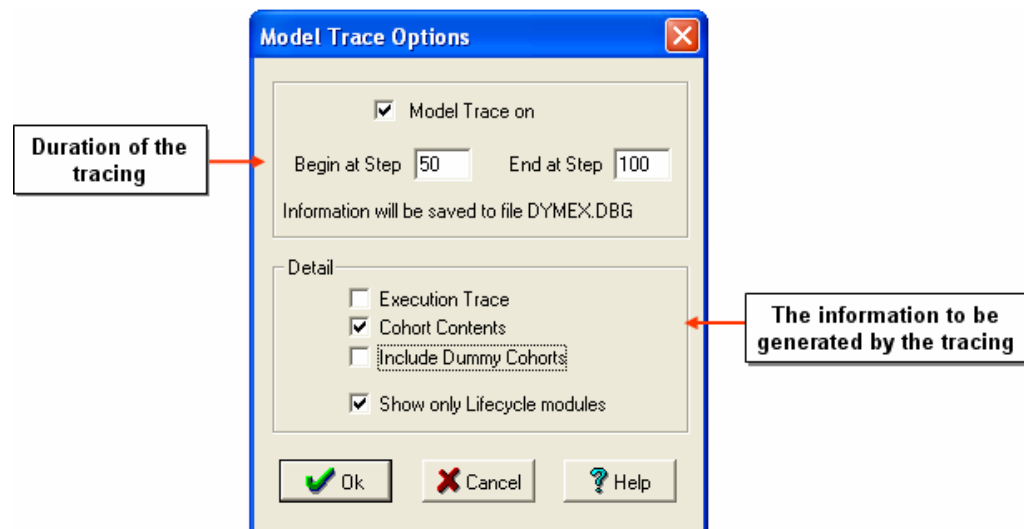
The **Model Trace** option instructs the *Simulator* to report on the status of various modules during a simulation run, and writes values of selected variables and other information to a Trace file (dymex.dbg). Tracing can be useful if you are interested in what is happening in individual cohorts within a lifestage. It can be useful to advanced users, who clearly understand the internal operation of DYMEX, in helping to optimize process parameters. However, it is also of use to novice users who are struggling with these concepts, in that they can follow exactly what happens within the cohorts (see, for example, Figure 5-1, which is Model Trace output that clearly demonstrates the movement of individuals through cohorts). Use of this option can slow a simulation run, as well as generating large files of information. For this reason, trace output is always disabled when a multiple simulation (Sequence) is being run.

To turn on the tracing facility, open the **Model Trace Options** dialog (Fig. 25-2), select **Model Trace on** option and type the beginning and end timestep for the tracing period. The **Detail** section enables different trace information to be selected, as follows.

- **Execution Trace** – reports in detail on creation, updating and destruction of each cohort. It can generate a very large amount of information. This option would be invoked only rarely, in cases where an error is suspected in the operation of the *Simulator*.
- **Cohort Contents** – reports the state of each cohort (including the value of each Cohort Property) during each step of the simulation. This can be a very useful option to check the detailed operation of processes when the output available from the graphs and tables is not detailed enough.
- **Include Dummy Cohorts** – when used in conjunction with the previous option, includes the state of “dummy” cohorts in the report (see *Dummy Cohorts*, page 33).
- **Show only Lifecycle modules** – disables the generation of trace output for any module except those of type Lifecycle. In the many cases when only the cohort contents are of interest, this option can reduce the size of trace files considerably.

The format of trace output is described in detail in Appendix 1.

Fig. 25-2 A Model Trace Options dialog box.



## 25.3 Current Model Options

The **Current Model** preferences relate to the particular model that is currently loaded, and these settings are saved in the Simulation file (see *DYMEX model files*, page 10). The **Model Options** dialog is shown in Fig. 25-3.

The **Run Description** is a line of text that is used as a heading for graphs, maps, tables and files that show results from a run of the model. The required text is typed into the **Run Description Format** text box. A number of special text strings (macros), all beginning and ending with angle brackets (<>) are available, and can be placed anywhere within the Run Description Format text. A macro can be selected for insertion at the current cursor position by clicking on the selection button (▶) and then selecting the required macro from the drop-down menu. When the model is run, these macros are expanded as described in the table below:

Macro	Expands to
<RunId>	The count of runs in the current <i>Simulator</i> session
<Date>	The date on which the current simulation was started
<Time>	The time at which the current simulation was started
<Model>	The full name of the currently loaded model
<Parameters>	The full path of the parameter file used for the current run
<Sequence>	The name of the Run Sequence in use for this run, or blank if this is a “Single” run
<MName>	The initialization setting (as shown in the Model Components window) for the module with name “MName”

For example, the Run Description Format

**Run <RunId>, dated <Date>, <Time> using Met Data <Meteorological Data File>**

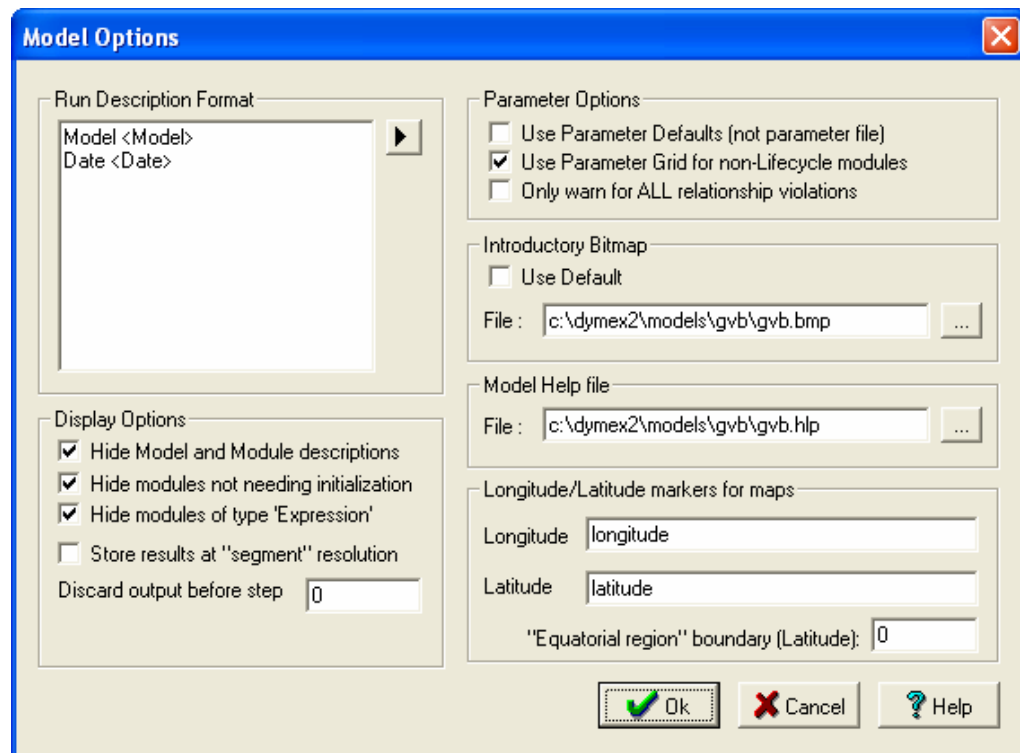
would expand to something like the following line for output:

**Run 3, dated Mar 16 1998, 14:56 using Met Data c:\data\ctowers.dat (1 Jan 90)**

**Display Options** specify some options used in the display of model information either in the Components Window or in tables or charts.


- **Hide Model and Module descriptions** - specifies that any descriptions of the model and the modules provided by the model designer are not to appear in the Component Window. It is advisable not to include these descriptions most of the time to obtain a less cluttered access to model operations, but they should be included when the Component Window is to be printed.
- **Hide modules not needing initialization** - specifies that modules that do not require any user initialisation are not to appear in the Model Components window. This is a useful option for large models as it can reduce screen clutter and concentrate the user's attention on the most significant model features.
- **Hide modules of type "Expression"** - specifies that **Expression**, **Equation** and **Function** modules are not to be shown (even if initialisation is available for them). This option would be used only after the affected modules have been initialised.
- **Store results at "segment" resolution** - if a model uses "segmented" timesteps (see the discussion of *Model Timestep* in the *Builder User's Guide*), this option specifies that results are to be stored at the most detailed (segment) timestep rather than the basic (1-day) timestep.
- **Discard output before step** - here the model user can choose to discard the results from a specified number of timesteps at the beginning of a simulation. Usually a model run needs to "settle down" from the supplied initial conditions and hence there is some advantage in not storing those results (especially in automatic "multiple" runs).

**Fig. 25-3 Model Options dialog showing use of an introductory bitmap and model-specific help file**



The **Parameter Options** panel specifies a few options for dealing with parameters.

- **Use Parameter Defaults (not parameter file)** - specifies that the model is to use the parameter defaults set in the *Builder* (and not access a Parameter File at all). This can be useful while a model is being constructed, as it avoids problems with parameter changes made in the *Simulator* not being transferred to the GMD file (or conflicting with settings in the *Builder*).
- **Use Parameter Grid for non-Lifecycle modules** - when ticked, this option indicates that the Parameter “Grid” view should be used in preference to the Parameter “Tree” view for viewing parameters (see *The Parameter “Grid”*, page 81). Note that the Parameter Grid is never used for Lifecycle modules nor for modules that use sub-populations.

The user can specify a bitmap, which is shown while the model is being loaded (instead of the default one that the *Simulator* shows normally). The **Use Default** box must first be unchecked, and then the name of the bitmap file can be either typed into the corresponding edit box, or the **Browse** button (  ) can be used to find it.

The **Model Help file** panel is used to specify the location of a “Help” file containing information specific to the model. The file must be in the standard Microsoft Windows® Help (.hlp) file format. The Help file is then available from the *Simulator Help* menu under **Currently loaded Model**.

The **Longitude/Latitude markers for maps** panel allows the user to specify two strings which identify the DYMEX (summary) output variables that describe the longitude and latitude (in decimal degrees) for the locations being run. The first summary variable that contains the specified longitude marker (case is ignored) is considered to be the “Longitude” variable (and similarly for latitude). These variables will then be automatically used as the coordinates when data is plotted on a map. If these fields are left blank, no maps can be produced for the model. As an example, if the model’s longitude and latitude variables are named “*Longitude (degrees)*” and “*Latitude (degrees)*”, respectively, any of the following pairs of markers might be acceptable choices:

“*Longitude (degrees)*” and “*latitude (degrees)*”  
 “*longitude*” and “*latitude*”  
 “*long*” and “*lat*”

Note, however, that if the last set were used, and a summary variable with the name of (say) “*Lateral Extension*” was also present in the model, there would be a possibility that the wrong variable would be used for latitude.

The “**Equatorial region**” **boundary (Latitude)** defines a longitudinal band around the earth, centred on the equator and extending the specified number of degrees north and south. Some modules use this for their calculations (notably the **Climate Change Scenario** and **CLIMEX Match Climates** modules). Users should refer to the descriptions of these modules for more details.

## 25.4 The Hints Window

The Hints window displays hints and explanations, which relate to the window or dialog that is currently active. If present, it is always the top window, and thus always visible. Whether or not it shows by default at the start of a *Simulator* session is set in the **Operating Preferences** dialog (see *Operating Preferences*, page 83). It can be removed temporarily by clicking on the “Close this ‘Hints’ Window” button in the Hints window itself. It can be removed or restored at any time by clicking on the **Show Hints** menu option in the **Preferences** menu.

## 26 Running the Model



Once the model has been successfully initialized (indicated in the Model Components window by a tick in front of every module symbol), a simulation can be run.

### ➤ To initiate a simulation run

- 1.** Make sure an appropriate **Run Identifier** has been specified to uniquely identify the run on charts, tables and files (see *Current Model Options*, page 86).
- 2.** Decide whether a **Single** or **Multiple** run type is to be performed. A single run performs a single simulation for the specified period. A multiple run runs the model several times, with the number of runs and their settings controlled by a **Run Sequence** (see *Run Sequences*, page 94).
- 3.** If a **Multiple** run is required, specify it by either using the Quick-selector buttons in the Component Window and then selecting **Execution|Run** from the menu (or menu bar) or select the **Execution|Multiple run with sequence** from the menu and then select the required sequence from the resulting menu or dialog.
- 4.** If a Single run is required, select **Execution|Run** or left-click the **Run** button (⚡) on the Button Bar.

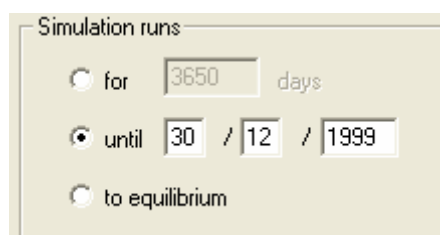
### 26.1 Running to Equilibrium



As an alternative to running simulations for a certain number of days, a model can be run until it reaches a user-defined equilibrium state (Fig. 26-1). “Equilibrium” is defined to occur when the results for successive years are “identical”, with the exact definition of “identical” supplied by the user. With some models, an equilibrium run will not be possible, and this will be indicated by the **to equilibrium** option in the Timer initialization dialog being disabled. An equilibrium run is possible only if

1. The model designer has allowed for this capability by linking an “Equilibrium Variable” to the **Timer** input
2. All modules are capable of reaching an equilibrium state
3. An equilibrium state has been defined by the model user

**Fig. 26-1 A section of the Timer initialization dialog box.**





The first condition is determined by the structure of the model, and can be checked by clicking on the **Timer**'s **Module Information** option. Only if an input variable for the **Timer** is shown, will an equilibrium run be possible.

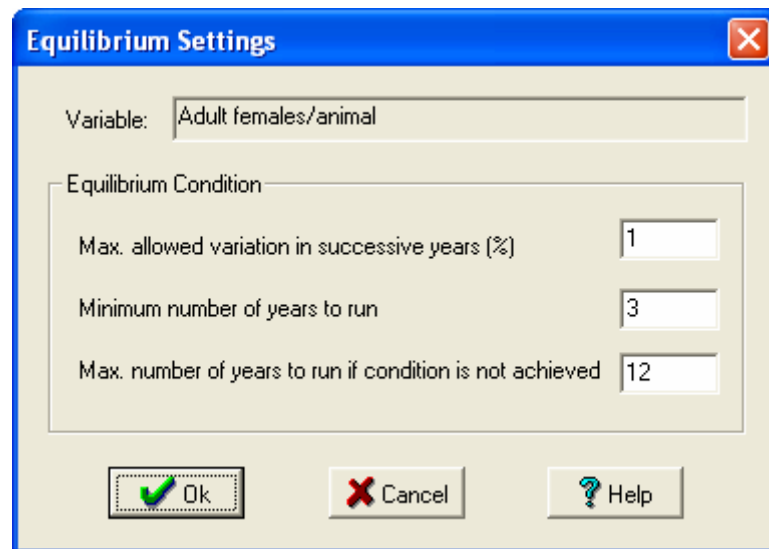
The only types of modules that may not be capable of reaching an equilibrium state are the **DataFile** and **MetBase** modules. Obviously, if these are reading actual data, there is no way that equilibrium can be guaranteed. Equilibrium runs in models using these modules are possible if only a single year of data is used, and reused for successive years. This is conveniently achieved by selecting the **Re-use first year's data for successive years** option within the Data File Format dialog (the option is found under the **More Options** button in that dialog).

☒ Re-use first year's data for successive years

## 26.1.1 Defining the Equilibrium

Before a model can run to equilibrium, an equilibrium state must be defined. To do this, select **Execution** from the main menu bar and select the **Equilibrium Settings** option. This displays the **Equilibrium Settings** dialog, where the equilibrium state is defined (Fig. 26-2).

**Fig. 26-2** Equilibrium settings with example equilibrium settings.



**Equilibrium Variable** The **Equilibrium Variable**, which will determine whether equilibrium has been reached, is shown at the top of the dialog. This variable has been selected by the model designer and cannot be changed without modifying the model in the **Builder**.

Equilibrium is determined by comparing the equilibrium variable's state between two successive years. The year's total for the variable (obtained by adding the variable's value for each timestep over a year) is compared with the previous year's total. If the difference between these two quantities is less than

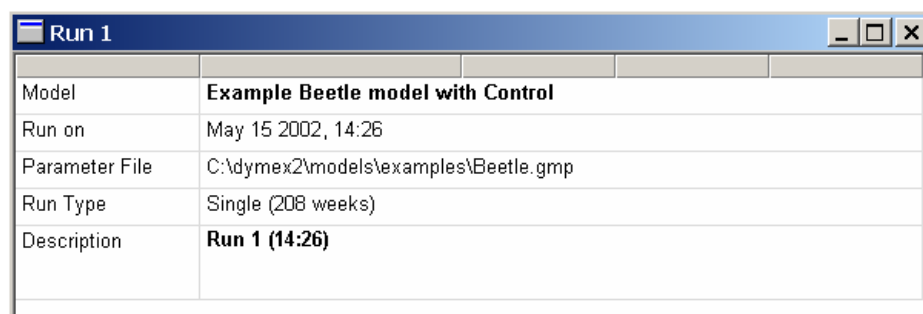
or equal to the value defined in the **Max allowed variation in successive years (%)**, the system is considered to be at equilibrium.

Use the **Minimum Number of Years to run** and **Max. number of years to run if condition is not achieved** to set limits for the run duration. The former sets the number of years the simulation runs before the equilibrium condition is checked for the first time. The latter sets the maximum duration of the run in case equilibrium is not achieved.

## 26.2 The Run Window

**!** At the conclusion of a simulation (whether a single run, or a run sequence), a **Run Window** is displayed. The Run Window is labelled with the sequential number of the run (for example, “**Run 5**”), and displays information about the run. This information includes the name of the model, the parameter file used, the **Run Description** (see Section 25.3, page 86), the type of run (i.e., whether single or multiple) and the run duration. If one or more **Run Summary** variables have been defined and the run just completed was a **Single** run, the Run Window will also display a table that lists each Summary Variable and its value. All operations that relate to the particular run must be performed from the **Run Window**. For example, tables, charts, maps and files can only be created from the Run Window, as they need the results from a particular run. Results (except for run summaries) are kept only for the most recent run. Therefore, charts, tables and graphs cannot be produced for runs other than the most recent run, even if the run window for the run is still open. All Run Windows except that relating to the last run are automatically minimized. The Run Window can be printed from the **File | Print** menu. Below is an example of a Run Window without run summary variables defined (Fig. 26-3). An example of the Run Window for a model that has such variables defined can be seen in Fig. 28-1.

**Fig. 26-3 An examples of the DYMEX ‘Run Window’ for a model that does not have any Run Summary variables defined.**



Model	Example Beetle model with Control
Run on	May 15 2002, 14:26
Parameter File	C:\dymex2\models\examples\Beetle.gmp
Run Type	Single (208 weeks)
Description	Run 1 (14:26)

## 26.3 Run Summaries

Run summaries enable the results of different runs to be compared by providing a summary of selected variables. Since the detailed results of individual runs are not directly available from *multiple* runs (i.e. simulations that use Run Sequences), Run Summaries are necessary for this type of simulation. *If a run sequence is run without a run summary, no results are shown.*

Run Summary Variables are created in the **Builder** (see *Builder User's Guide*, Section 11) and cannot be modified in the **Simulator**. To see what Summary Variables have been defined for the model, select **Summary Variables** from the **View** menu. This shows the **Summary Variables** dialog (Fig. 23-1). See *The SummaryManager module*, page 74 for a description of this dialog.

## 27 Run Sequences

Run sequences in DYMEX allow a set of simulation runs that vary in a systematic way to be run automatically. For example, Sequences can be defined that change parameter values systematically between runs (for sensitivity analyses), while other sequences allow the application and timing of events to be changed between runs (for example, to optimize management strategies such as pesticide applications). Sequences need to be defined by the user and named, after which they can be used in **Multiple** runs.

The following types of Sequences are currently available in DYMEX:

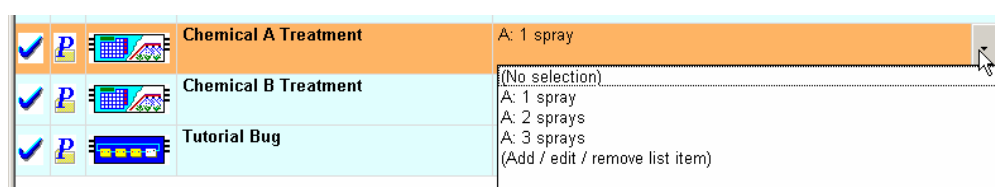
Event	change event application systematically between runs
DataFile	automatically run the model for a set of data files
MetManager	automatically run the model for a number of locations
Parameter	change parameter values systematically between runs
Compound	run the model for a combination of sequences
Simple	run the model a number of times (no changes between runs)

The **Event**, **DataFile** and **MetManager** Sequences are associated with instances of the corresponding modules. If multiple instances of one of these modules are present in a model, there will be a Sequence type associated with each module instance. For example, if a model contains two Event modules named “*Chemical Treatment*” and “*Vaccination*”, then separate “Event” sequences can be set up for each of these modules. Modules that can have Sequences associated with them have a “Quick Selector” button at the far right of their panel in the Component Window. Clicking on this button opens a “drop-down” window (Fig. 27-1) that lists the sequences defined for this module and allows one to be selected or edited. Note that the button is not available if no summary variables are defined for the model.



***Run sequences are of little use without first defining run summary variables. Read the appropriate section in the **Builder User’s Guide** (Section 11, Summary Variables) to see how to add summary variables to your model.***

**Fig. 27-1 Event module panel in Component Window, showing the drop-down window for Sequence selection.**



➤ **To create a run sequence**

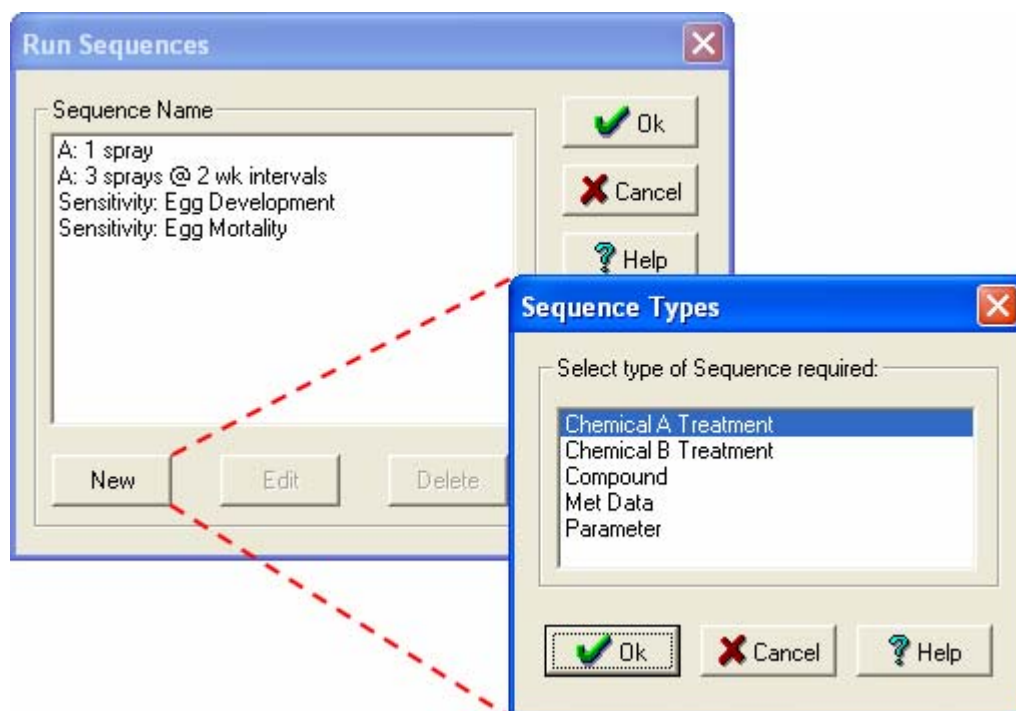
Either:

1. Select **Execution | Define 'Run Sequences'** from the **Model Component** window menu. This opens the **Run Sequences** dialog, which lists all the sequences that have been defined for this model.

or (if the required run sequence is associated with a module):

1. Click on the "Quick Selector" button at the right of the module panel (in the **Model Component** window) for which the new sequence is required and select "**(Add/edit/remove list item)**" from the drop-down list. This opens the Run Sequences dialog, listing all the sequences defined for the corresponding module.
2. Left click on the **New** button. This opens the **Sequence Types** dialog box (Fig. 27-2) if you have used the first method to start creating the sequence. If you used the second method, continue with step 4.
3. Choose the type of run sequence required.
4. The appropriate **Sequence Definition** dialog will open, where the exact form of the sequence can be defined.

**Fig. 27-2 The Run Sequences dialog box with an open Sequence Type dialog box in response to a click on the New button.**



Sequences that are associated with modules (such as **Event** sequences) can be conveniently selected, created or edited by using the "Quick Selector" button that will be present at the far right of the module's panel in the Component Window (see Fig. 3-4).

## 27.1 Event Run Sequences

Event Sequences are used to answer either of the following questions: (1) when is the best time to apply treatments, given a fixed number of treatments and fixed intervals between them, or (2) how many treatments are optimal, given a fixed starting time and intervals between treatments. The specified treatments are applied each year (with the exception of a nominated number of years at the start) during every simulation run in the sequence.

The two questions above correspond to the two types of Event run sequences that can be defined in DYMEX:

- **Vary Starting Date, fixed No. of Actions.** Using this sequence, we can find out the *best time* to apply a given treatment regime as illustrated in Example 1: We have a pasture that is attacked by whitegrubs and we want to apply 2 insecticide treatments at 3-week intervals and we want to identify the best time during the year to apply the treatments at a given location.
- **Vary No. of Actions, fixed Starting Date.** With this type of sequence, we can identify the *optimal number* of treatments per year, based on some criterion such as cost effectiveness. In Example 2, we want to apply insecticide treatments, starting on March 15, and spaced 3 weeks apart, and we want to know how many treatments would give us an optimal level of control.

With either type of Event sequence, we start by defining the desired attributes of a **Base Event**. In Example 1 this would be two treatments at three-week intervals, triggered on January 1. The Base Event for Example 2 could be two treatments at three-week intervals, triggered on March 15. To complete these sequences, the rules for changing the Base Events are then defined. In Example 1, the day on which the event is triggered might be incremented by 7 for each of a total of 52 runs to give weekly results for one year. In Example 2, we might add an extra treatment for each run, until we reached 12 treatments (a total of 11 runs).

### ➤ To set up an Event run sequence

1. In the Event Sequence Definition dialog (Fig. 27-3), enter a descriptive **Name** for the Event Sequence – this name will appear in the list of Run Sequences for the model. For Example 1 above, we might choose a name such as “2 sprays @ 3 wk intervals – vary start”. If you have more than one event type in the model (for example two different chemical treatments), you may want to include the event type in the name (for example, “Chem A: 2 sprays @3wks – vary start”).
2. Enter a short (1-6 letter) mnemonic that can be used as an event type label. By default, DYMEX uses the name of the Event module shortened to 6 characters. You should replace this with something more useful (eg, “H/A” for “Herbicide Application” or “Vacc” for “Host Vaccination”. This label will be used as a column label in tables to identify the treatment type.

3. Select the type of sequence required by clicking on the appropriate button in the **Sequence Variation Type** box.
4. In the **Base Event Definition**, describe the **Base Event** by filling in the text boxes appropriately, as described below.
  - **No. of actions in group** - is the total number of *Actions* within the one Event. In Example 1 above it is two, indicating that the spraying will be repeated once after the first trigger.

**Fig. 27-3 A “Run Sequence – Event” dialog box.**

The screenshot shows a dialog box titled "Event Sequence - Chemical A Treatment". It contains several sections with input fields and buttons. Annotations with arrows point to specific parts of the dialog:

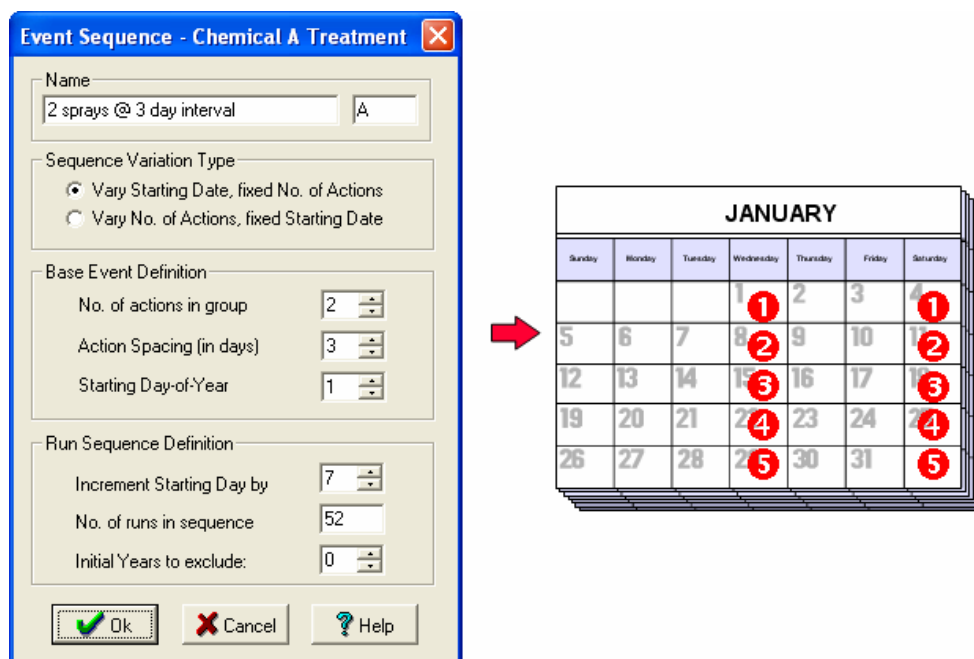
- Sequence type label:** Points to the "Treat" button next to the "Name" field.
- Type of event sequence variation:** Points to the "Sequence Variation Type" section, which has two radio buttons: "Vary Starting Date, fixed No. of Actions" (selected) and "Vary No. of Actions, fixed Starting Date".
- Replicates of events:** Points to the "No. of actions in group" field in the "Base Event Definition" section, which has a value of 2.
- Starting day increment and number of runs:** Points to the "Increment Starting Day by" field in the "Run Sequence Definition" section, which has a value of 7.

Other visible fields include "Name" (2 sprays @ 14 day interval), "Action Spacing (in days)" (14), "Starting Day-of-Year" (1), "No. of runs in sequence" (52), and "Initial Years to exclude" (2). At the bottom are "Ok", "Cancel", and "Help" buttons.

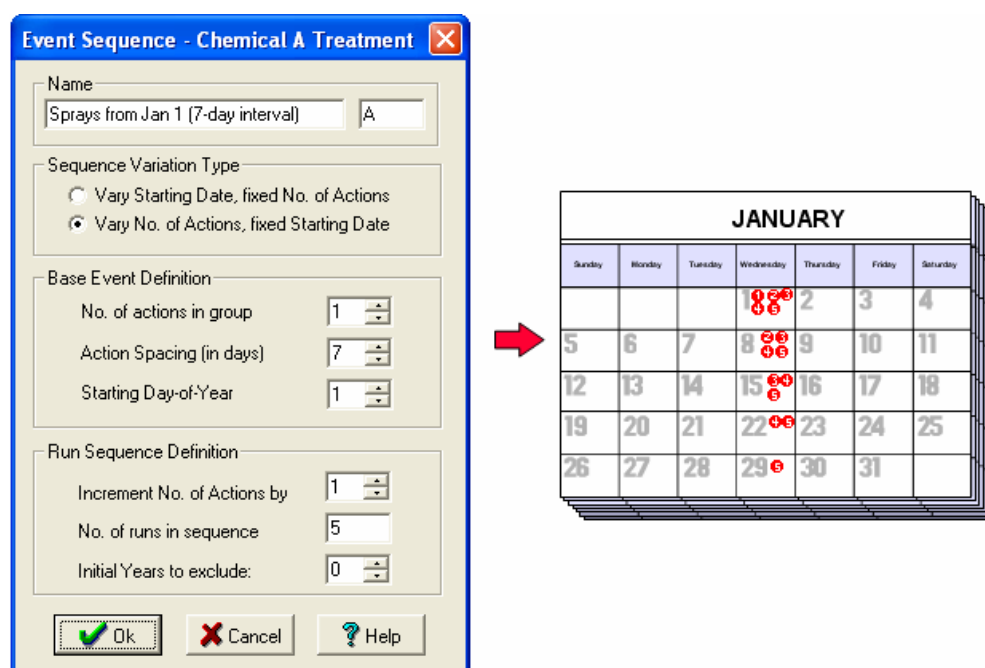
- **Action Spacing (in days)** - is the number of days between repeats of the *Action* (three weeks in Example 1).
  - **Starting Day-of-Year** - is the day of the year at which the Base Event in the run sequence occurs. In Example 1, this is the first day of the year.
4. In the **Run Sequence Definition**, describe how the Base Event is to be modified between successive runs in the sequence. This part of the dialog will change slightly, depending on which type of sequence has been selected in Step 2:
    - **Increment Starting Day by** is used for “Vary Starting Date...” event run sequences (Fig. 27-4). In each run, the starting date for the Event is incremented by the specified number of days.

- **Increment No. of Actions by** is used for “Vary No. of Actions” event run sequences (Fig. 27-5). In each run, the number of Actions is increased by this number, which will usually be set to 1.

**Fig. 27-4 A “Vary Starting Date, fixed No. of Actions” event Run Sequence (Example 1) with 2 Actions at 3 day intervals starting on Jan 1, giving the pattern of events shown on the calendar, with events in successive runs shown as ①, ②, etc.**



**Fig. 27-5 A “Vary No. of Actions, fixed Starting Date” event Run Sequence with an Event starting on Jan 1, consisting of one action for the first run (①), then adding an extra action (separated by a 7 day interval) for each successive run, for a total of 5 runs.**





- **No. of runs in sequence** is the total number of runs in the sequence. In Example 1 above this number is 52, which means that there are 52 runs with the Event being triggered one week later in each successive run. That is, sequential runs will trigger the Event on successive weeks of the year so that the results for each week can be compared.
- **Initial Years to exclude** specifies the number of years at the start of each simulation during which the specified treatment is NOT applied. Enough years should be specified here to allow the simulation to overcome starting conditions before treatments are applied.

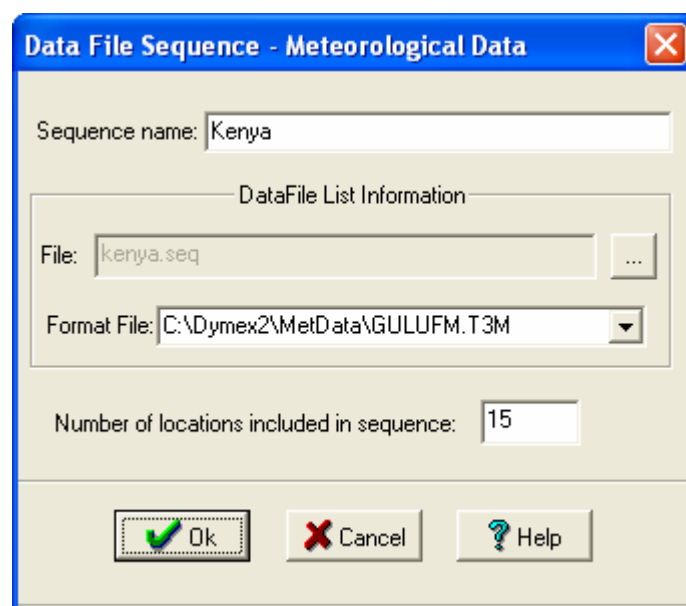
In Fig. 27-5, there is one action within the Base Event therefore there is only one action in run one. Since the **Action Spacing (in days)** is seven and **Increment No. of Actions by** is one, there are two events in run two, one on day one and one on day eight. This will continue until run 5 where there would be an event action on each week of the month.

Fig. 27-4 is an example of a Run Sequence with an Event with two actions separated by three days occurring in each run in the sequence. The calendar shows the dates on which event actions occur in the first 5 runs in the sequence, with the numbers in the circles indicating the simulation run number.

## 27.2 DataFile Run Sequences

Data File Run Sequences are used to automatically run simulations for a number of geographical locations, with the data for each location read from a **DataFile** or **MetBase** module. If latitude and longitude information for each location is also available, the results of such a set of runs can be plotted on a map (see, for example, Fig. 28-20).

**Fig. 27-6 A Data File Sequence based on the file “Kenya.seq”**



The **Data File Sequence** dialog (Fig. 27-6) can be accessed via the “Quick Selector” button at the right of the module’s panel in the Component Window (select **Add/Edit/Remove list item**). The name of the sequence must be supplied in the edit box at the top, while the name of the file (the *Sequence File*) that supplies the list of file names (one for each run in the sequence) can be obtained using the **Browse** button. The **Number of locations included in sequence** field may be left blank or set to 0 (in which case the sequence will include of all the files specified in the *Sequence File*), or it can be set to a positive value so that the sequence will consist of the corresponding number of runs only (if more files are named in the *Sequence File*, they will not be used). The **Format File** field is used to specify an “example” file whose format will be used for interpreting the data in each of the files included in the *Sequence File*. Clicking on the small button at its right side causes a list of suitable files (those whose format is known to the *Simulator*) to appear, from which the required file name can be selected. If no file with a suitable format is listed, the format of one file (not necessarily included in the sequence) must be set up in the **Model Components** window before the Sequence is set up (see Section 6.1.2, *Initializing Data File Modules (Basic Options)*).

An example of a *Sequence File* is shown below. The file is simply a list of file names, one file per line. *Sequence Files* must be prepared using a text editor such as *Notepad* (if a word processor such as *Microsoft Word* is used, the file must be saved in text only format). If *Notepad* is used, make sure when saving that the “**Save as type:**” option is set to All Files, otherwise Notepad will append the extension “.txt” to the filename, even if you have specified “.seq”.

```
c:\metdata\kenya\eldoret.dat
c:\metdata\kenya\embu.dat
c:\metdata\kenya\kabete.dat
c:\metdata\kenya\kapenguri.dat
c:\metdata\kenya\kisii.dat
c:\metdata\kenya\kisumu.dat
c:\metdata\kenya\kitale.dat
c:\metdata\kenya\malindi.dat
c:\metdata\kenya\marienne.dat
```

Either columnar (e.g., .dat or .txt) or comma-delimited (.csv) files are acceptable, but each file listed in the *Sequence File* must have exactly the same data format, so these file types cannot be mixed in the same *Sequence File*.

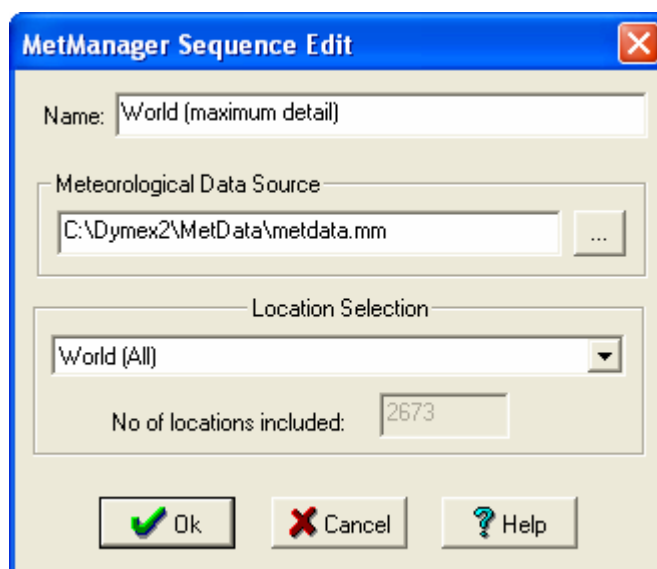


The most convenient way to provide latitude and longitude to the simulation, and to ensure that the correct values of these variables are used for each location, is to provide them as part of the **DataFile**’s header. For this to work, a **QueryFile** module must be the module type being used to read the values of latitude and longitude. Make sure that the same position in the file header is used for these variables in each file constituting the Sequence. Then use the **Read value from a data file** option (Section 9.2) when setting up the **QueryFile** (before running the Sequence).



## **27.3 MetManager Run Sequences**

MetManager Run Sequences are used to automatically run simulations for a number of geographical locations using long-term average weather data, with the data for each location read from a **MetManager** module. The MetManager database includes latitude and longitude information for each location and therefore the results of such a set of runs can be plotted on a map. The MetManager obtains its data from a MetManager database (which can be either a Microsoft Access® MetManager (.mm) file or a .bmt file as used with Version 1 of the CLIMEX program). See Appendix x for more information about the MetManager database and program. Sets of locations within the database are accessed using “Location Selections”. A “Location Selection” is a set of criteria (such as country name, minimum and maximum latitude), and all locations in the database that meet those criteria can be extracted using this selection. A Microsoft Access® MetManager (.mm) file stores all its “Location Selections” within it as database “Queries”, while those for .bmt files are stored as “Selection Files” (.sel).

**Fig. 27-7 The MetManager Sequence Edit dialog**



### ➤ **To set up a MetManager run sequence**

- 1.** Open the **MetManager Sequence Edit** dialog by clicking the “Quick Selector” button at the right of the MetManager module’s panel in the Component Window, and selecting **Add/Edit/Remove list item** from the drop-down menu (in Fig. 27-7).
- 2.** Select the Meteorological database by clicking on the **Browse** button (  ) and then navigating to the file, or type its directly into the edit field.
- 3.** Click on the drop-down button (  ) in the **Location Selection** panel. This will display a list of available “Location Selections” for the database chosen in step 1. Choose the required “selection” from the list. If you have chosen a .bmt file for the database, and no *Location Selections* appear in the list, make sure that the “Location Selection” directory is correctly set in **Operating Preferences** (see Section 25.1).

4. The name of the chosen *Location Selection* is automatically used as the Sequence name. If this is not suitable, change it to a different name in the **Name** edit box.
5. Click **Ok** to complete the Sequence setup.

Note that the number of locations in the database that fit the *Location Selection* criteria is shown in the **No. of locations included** field.

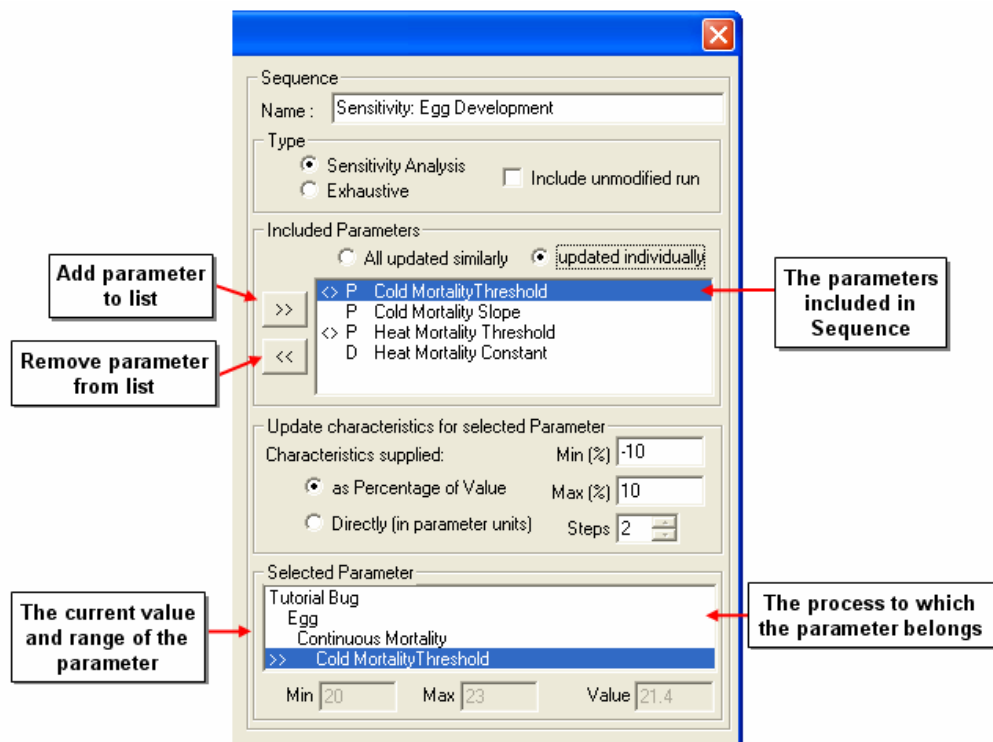
## 27.4 Parameter Run Sequences

Parameter run sequences can be used to perform sensitivity analyses on the model. The values of one or more parameters can automatically be varied systematically between successive runs, and the effect this has on the model can be presented in tables. For example, the effect of varying mortality parameters in a lifestage on the total population can be ascertained.

Name  
Adding and  
removing  
variables

The **Parameter Sequence** dialog is accessed by selecting “Parameter” in the **Sequence Types** dialog (Fig. 27-2). (The **Sequence Types** dialog is opened by selecting **Execution | Define ‘Run Sequence’** from the menu, and selecting **New** in the **Run Sequence** dialog). The **Parameter Sequence** dialog lists the model parameters on the left, while the sequence is defined in the right-hand side panel (Fig. 27-8). The parameter list is set out in exactly the same format as the parameter list in the Parameter window (see *The Parameter “Tree” Window*, page 77).

**Fig. 27-8 Sequence section (right side) of the Parameter Sequence dialog box with an example sequence.**



The dialog has the name of the sequence at the top. A unique name must always be supplied for a sequence, so that it can be identified in the list of sequences when required for a simulation.

The **Type** panel allows the user to choose how the Sequence is to cycle through the parameter values, and whether the Sequence should include the unmodified run (i.e., the run when all parameters take their unmodified value). Two Parameter Sequence “types” are available:

- **Sensitivity Analysis** – one parameter is cycled through its variations while all other parameters are kept at their unmodified values. This is repeated for each parameter. Thus the total number of runs in a Sensitivity Analysis is given by

$\sum n_i$ , where  $n_i$  is the number of **Steps** selected for the  $i$ -th parameter.

- **Exhaustive** – All combinations of all of the parameter’s variations are used in the Sequence. This can produce a large number of runs (the product of all the  $n_i$ ), even if only a few parameters are included.


The **Included Parameters** panel is where the user specifies how the individual parameters selected for the sequence (shown in the panel’s list) are to be varied between runs in a sequence. They can all have their values changed in the same manner, or each parameter can have its own change procedure. The required option is indicated using the buttons above the **Included Parameters** list.

If **All updated similarly** has been selected, then the specification will apply to every parameter in the **Included Parameters** list box. Otherwise, (**updated individually**) it applies only to the currently selected parameter. The buttons and text-entry boxes under the parameter list allow the user to specify the range and steps for the proposed variation in the parameter or parameters.

At the bottom (**Selected Parameter**) is a text box that shows the context (the module and, if it is a Lifecycle parameter, the lifestage and process) of the parameter that is currently selected in the **Included Parameters** list. It also lists, for information, the current value and minimum and maximum allowed values for the parameter.

➤ **To specify a Parameter sequence**

1. In the Parameter Sequence dialog, enter a descriptive **Name** for the Parameter Sequence – this name will appear in the list of Run Sequences for the model. For example, a name such as “*Development Parameter Sensitivity (20%)*” could be used.
2. Decide whether the parameters will all be changed in the same way for the sequence (i.e., all may be adjusted in the range “value”–10% to “value”+10%), or whether separate rules will be applied to each parameter. Select either **All updated similarly** or **updated individually** to reflect that choice.

3. Choose a required parameter from the **Model Parameters** list and left-click the **Add** button () or double-click on the parameter name in the **Model Parameters** list. This will add the parameter to the **Included Parameters** list at the right of the dialog. Repeat this for all the parameters that are required.
4. If **All updated similarly** has been chosen in Step 2, then provide the minimum (**Min (%)**) and maximum (**Max (%)**) values for the parameter in the sequence (as % deviations from its current value), and the number of steps (**Steps**). For example, if the parameter has a current value of 50, and the parameter's deviations are specified as -10% and 20%, with 6 steps, then it will take on the values 45, 48, 51, 54, 57 and 60 in the sequence.
5. If **Updated individually** has been chosen in Step 2, then, for each parameter in turn, select how its value is to be updated for the sequence from the **Update characteristics for selected parameter** box. For parameters that are updated **as Percentage of value** provide the minimum (**Min (%)**) and maximum (**Max (%)**) values for the parameter in the sequence (as % deviations from its current value), and the number of steps (**Steps**). For parameters that are updated **Directly (in parameter units)** provide the minimum (**Min**) and maximum (**Max**) values for the parameter in the sequence (as actual values), and the number of steps (**Steps**).

Note that the **Included Parameters** list shows the updating method used for each parameter (when individual updating is used) by a character preceding the parameter name. A “P” is used for parameters that are updated proportionally, while a “D” is used for those updated directly. The list also provides an indication of parameters that cannot be adjusted over the specified range. If a “<” character precedes the parameter name, then the specified sequence range would cause the parameter to be set to a value smaller than its smallest allowed value, while it would become too large if a “>” character is present. The sequence ranges need to be adjusted for any such parameter (using the **Max** and **Min** edit boxes) before the Parameter Sequence can be saved.

#### ➤ To remove a parameter from the sequence

1. Select the variable from the **Included Parameters** list and left-click the **Remove** button (.

The following examples illustrate Parameter Sequences using the “**All updated similarly**” and “**updated individually**” options, respectively. Both examples assume that **Sensitivity Analysis** has been selected and the **Include unmodified run** option has not been checked.

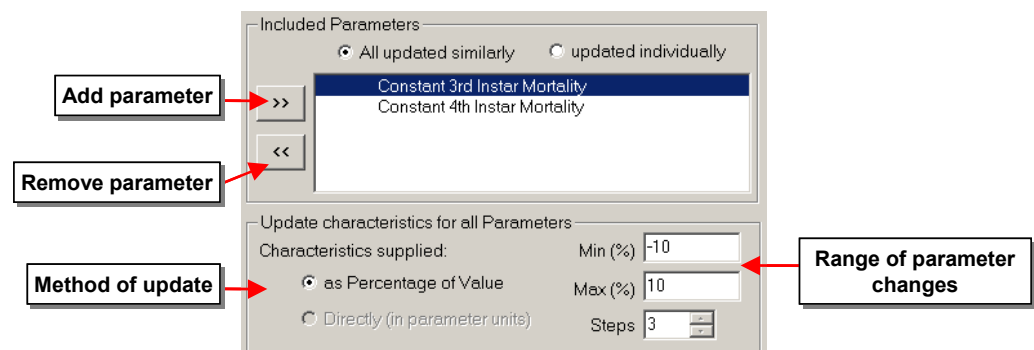
#### • Parameters “All updated similarly”

The **All updated similarly** option specifies that all parameters used in the Parameter Sequence will be changed in the same way within the sequence (i.e. the update characteristics apply to every parameter in the list). Note that with

this option, the “**Characteristics supplied: Directly (in parameter units)**” selection is disabled (it makes no sense here, since different parameters can have very different magnitudes). Thus the update characteristics (**Min** and **Max**) can only be supplied as percentages of the current value.

**Min (%) & Max (%)** **Min (%)** is how much the parameter’s current Value (**Value**, shown in the lower right of the dialog) must be adjusted (in %) to obtain the lowest value required for the sequence. If the lowest value is less than that the current value, then **Min (%)** should be negative, i.e. -10% of the default value of 0.1 is 0.09. Similarly, **Max (%)** is how much the parameter’s current Value must be adjusted to obtain the highest value required for the sequence. If the selected parameter has a range, the parameter value, after either the **Min (%)** or **Max (%)** deviations are applied, must still be within that range. Each parameter whose value would fall outside its permitted range is indicated with “<” and/or “>”, depending on whether the lower or upper bound is overstepped.

In the example in Fig. 27-9, two parameters have been added to the **Included Parameters** list. Values of -10% and +10% have been chosen for the minimum and maximum, respectively. The number of steps has been chosen as 3, hence this run sequence will perform 6 run sequences the first being -10% of the current value of the “3<sup>rd</sup> instar mortality” parameter. The last run in the sequence will use +10% of the default value of the “4<sup>th</sup> instar mortality” parameter. Table 27-1 shows the 6 simulations performed using this Parameter Sequence, along with the values of the two sequenced parameters used for each run. Note that the current values (0.1 and 0.07) are retained for the parameters that are not the subject of variation in a given run.



**Fig. 27-9 Example of an “All updated similarly” run sequence.**

**Table 27-1 The values of the parameters used in each run within the Parameter Sequence set in Fig. 27-9.**

<i>Run</i>	<i>3<sup>rd</sup> instar Mortality</i>	<i>4<sup>th</sup> instar Mortality</i>
1	0.09 (-10%)	0.07
2	0.1	0.07
3	0.11 (+10%)	0.07
4	0.1	0.063 (-10%)
5	0.1	0.07
6	0.1	0.077 (+10%)

- Parameters “Updated individually”

The **Updated individually** option allows each parameter’s sequence values to altered using a different setting for the maximum and minimum values and a different number of steps. In addition, the user has the option of specifying sequence limits directly rather than as a % offset from the current value. Different parameters included in the sequence can have different update methods.

Each parameter must be selected in turn by clicking on its name in the **Included Parameters** list-box. The update method to be used for that parameter can then be selected (note that if the parameter’s current value is 0, the proportional update method is not available). If the proportional update method (**as Percentage of value**) is selected for the parameter, its **Min (%)** and **Max (%)** limits for the sequence can be selected in the same way as described for grouped parameter updating in the previous section. If the **Directly (in parameter units)** method is chosen, the **Min** and **Max** values are the minimum and maximum values of that parameter to be used in the sequence. Note that the update method used for each parameter is shown as a “P” or “D”, respectively, just preceding the parameter name in the **Included Parameters** list-box.

With the **Directly (in parameter units)** update method the maximum and minimum values for the run sequence **MUST** be within the range possible for the parameters listed at the bottom right of the dialog box (Fig. 27-10).



**Fig. 27-10 Two examples of individually updated parameters, the first directly and the second as a percentage.**

The figure shows two side-by-side screenshots of a software interface for parameter management. Both windows are titled 'Included Parameters' and have radio buttons for 'All updated similarly' and 'updated individually' (selected). Below the title bar is a list box containing 'D Constant 3rd Instar Mortality' and 'P Constant 4th Instar Mortality'. Below the list box is a section titled 'Update characteristics for selected Parameter'. In the left window, 'Characteristics supplied:' has 'Min' set to 0.05, 'Max' set to 0.15, and 'Steps' set to 2. The radio buttons for 'as Percentage of Value' and 'Directly (in parameter units)' are both present, with 'Directly (in parameter units)' being selected. In the right window, 'Characteristics supplied:' has 'Min (%)' set to -50, 'Max (%)' set to 50, and 'Steps' set to 4. The radio buttons for 'as Percentage of Value' and 'Directly (in parameter units)' are both present, with 'as Percentage of Value' being selected.

In the example above (Fig. 27-10) the values for the six run sequences are listed below (Table 27-2).

**Table 27-2 The values of the parameters used in each run within the Parameter Sequence set in Fig. 27-10.**

<i>Run</i>	<i>3<sup>rd</sup> Instar Mortality</i>	<i>4<sup>th</sup> Instar Mortality</i>
<b>1</b>	0.05	0.07
<b>2</b>	0.15	0.07
<b>3</b>	0.1	0.035
<b>4</b>	0.1	0.058333
<b>5</b>	0.1	0.081667
<b>6</b>	0.1	0.105

## 27.5 Compound (Nested) Sequences

A **Compound Sequence** consists of two or more sequences, which are run together so that DYMEX runs all possible combinations of the two sequences. For example, assume we have a model that contains two chemical treatments (Chemical A and Chemical B), and the following sequences have been defined:

*3 Chemical A Treatments:* Treat 3 times at weekly intervals, with starting weeks for treatments in week 1, 2, ..., 52 (giving 52 runs).

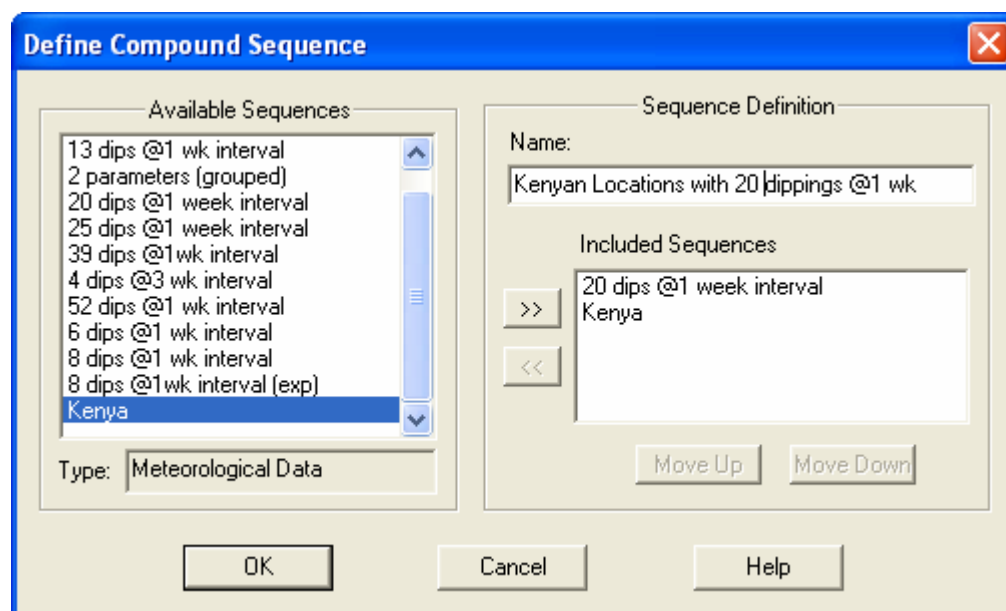
*4 Chemical B Treatments:* Treat 4 times at weekly intervals, with starting weeks for treatments in week 1, 2, ..., 52 (giving 52 runs).

By including these two runs in a nested sequence, DYMEX could simulate all combinations of 3 *Chemical A* treatments with 4 *Chemical B* treatments in a single run (in a total of 52×52 simulations).

➤ **To specify a Nested Sequence**

1. Make sure that all the sequences to be included in the nested sequence have been defined.
2. From the **Execution** menu, select **Define ‘Run Sequences’...**. This will open the **Run Sequences** dialog, which lists all the sequences that have been defined for this model. Click on the **New** button to open the **Sequence Types** dialog.
3. Select “*Compound*” as the type of Sequence required and click on the **Ok** button. This opens the **Define Compound Sequence** dialog (Fig. 27-11).

**Fig. 27-11 The Compound Sequence dialog.**



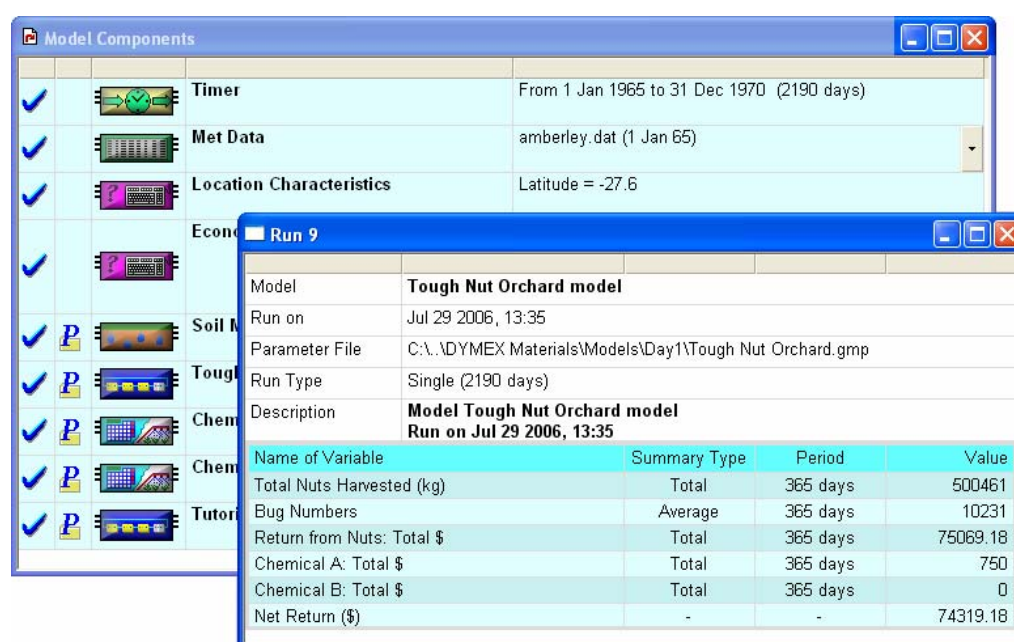
4. From the **Available Sequences** list at the left, select the first sequence to be included in the Compound Sequence. Note that the box labelled **Type**, at the lower left, will show the *type* of Sequence that has been selected (either the name of a module, or Parameter). Click on the selector button (**>>**) to add the sequence to the **Included Sequences** list.
5. Repeat Step 4 to add as many sequences as required to the list. Note that once a sequence of a particular type is included, another sequence of the same type cannot be added (Using the example given above, it is not permitted to include two sequences that both vary the time of application of *Chemical A*).
6. The included Sequences may be reordered as required using the Move Up and Move Down buttons.
7. Provide a name for the Sequence in the **Name** panel and then click on **Ok** to exit the dialog.

## 28 Displaying Output



Once a simulation has been performed and the **Run Window** is displayed, a presentation of the detailed results is possible. There are two ways to view the output from most simulations; either graphical using charts or in a table. Detailed results can also be written to a file or a database. For Multiple runs that include simulations from a range of geographical locations, the results can also be plotted on a map. *Note that any of these displays or files can only be created when a **Run Window** is the active window (with the displays relating to that run).*

Fig. 28-1 Main (Component) window with Run window in front.



Above in Fig. 28-1 the run window is present and selected (i.e. the bar along the top of the window is the “active window” colour – blue in this case) and the **Chart**, **Table**, **File** and **Database** creation buttons will be enabled in the tool bar.

### 28.1 Creating Charts



Charts provide an easy way to examine the results of simulations. Patterns that may be difficult to discern in a table can become apparent very quickly when plotted in a chart. Once a chart has been created, DYMEX provides a way to save the format of that chart, allowing quick loading of charts with the same format later. This allows multiple simulations with rapid viewing of results after each run. Charts can be copied to word processors and printed to local and network printers (*see below*).

➤ To create a chart

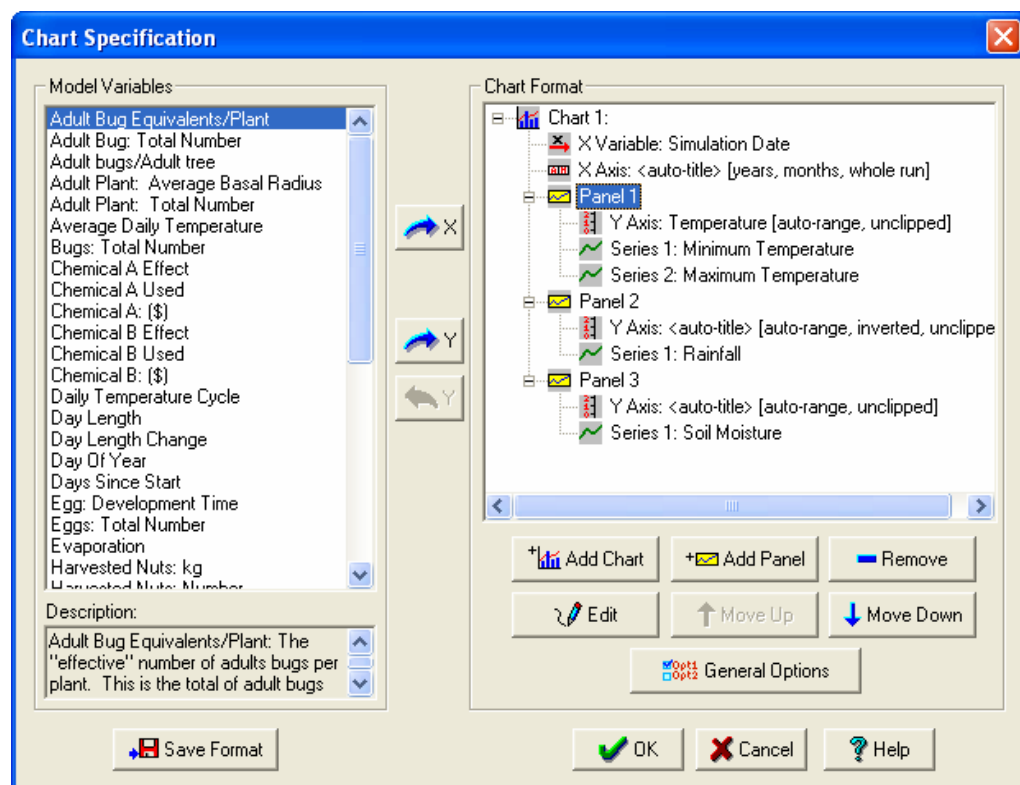
1. Select **Results** from the main menu and choose **Chart** or left click the chart button on the tool bar. If any chart formats have been saved, this will result in a small popup menu, giving a choice of chart formats. Select the **New** option from this menu to create a new chart, or select one of the saved chart formats to create a chart with the same format, but using the data from the current run.






Note that if you select a saved format, the chart is drawn without the Chart Setup dialog being shown. This behaviour can be changed by checking the “*Always show Display Dialog*” option in the Operating Preferences dialog (reached via the **Preferences|Operating** menu item).





### 28.1.1 The Chart Specification dialog

The Chart Specification dialog is used to set up one or more charts for plotting on a page. This dialog (Fig. 28-2) seems rather daunting at first glance, but with a little practise, users will quickly become familiar with its operation.

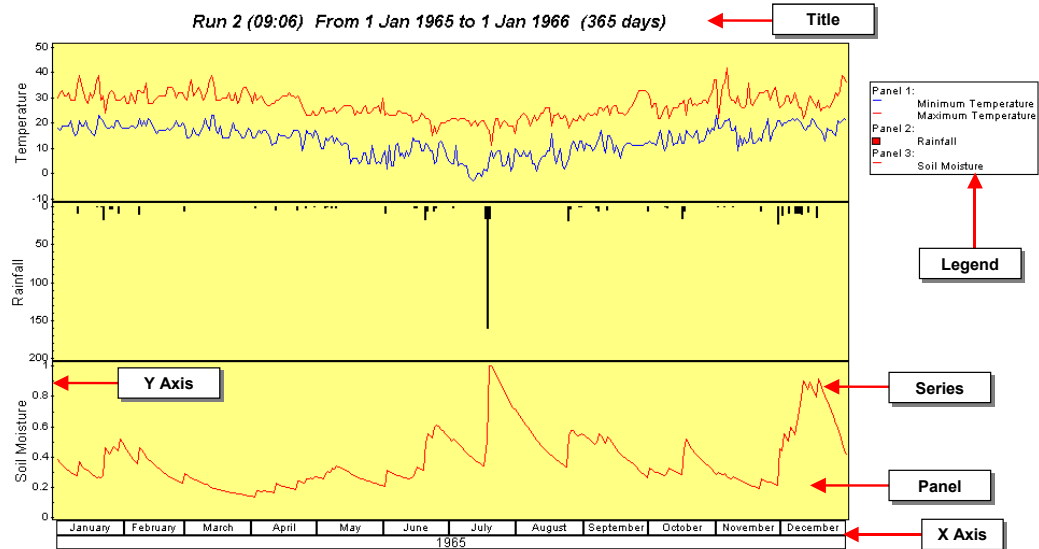
Fig. 28-2 The Chart Specification dialog.



All the model variables that are available for charting are listed in the panel at the left. The chart is defined in the panel at the top right (**Chart Format**). The format is organized in the form of a “tree” diagram. At the top level is the **Chart** (indicated by the symbol ) (and the associated **X Variable**, ). A Chart consists of an **X Axis**,  (and the associated **X Variable**, ) and a number of **Panels** (). Each panel can

contain a number of data **Series** (indicated as ,  or ), and one or two **Y Axes** (). The chart that would result from the format settings shown in Fig. 28-2 is shown in Fig. 28-3 for a 1-year run of the model. Some annotations have been added to show the major chart features.

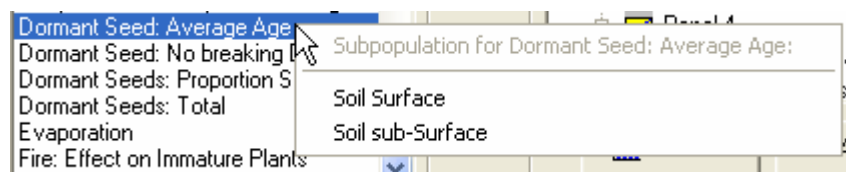
**Fig. 28-3** The chart resulting from the chart format settings shown in Fig. 28-2.



### ➤ To create a simple chart

1. Highlight the required x-axis variable in the **Model Variables** list by clicking on it and then click on the button labelled **X**. The x variable will usually be a **Timer** module output variable such as *Days since Start* or *Simulation Date*.
2. Select the chart's **X Axis** node by clicking on it and then click on the **Edit** button. This pops up the **X Axis Format** dialog from which various x-axis properties can be set. These are described in detail in Section 28.1.3.
3. Select a variable to be graphed from the **Model Variables** list and click on the button labelled **Y** to place it into the first panel. If the selected variable has subpopulation components, a popup menu will appear, from which the required subpopulation can be selected (Fig. 28-4).

**Fig. 28-4** The popup menu allowing selection of a subpopulation component when a variable with multiple subpopulations is selected for charting.



- 4.** As each variable is added, the **Series Format** dialog appears. In this dialog (described in detail in Section 28.1.6) you can set series properties such as style and colour.
- 5.** If more variables are to be plotted in the same panel, repeat steps **3** and **4** for each required variable.
- 6.** Select the panel's **Y Axis** node by clicking on it and then click on the **Edit** button. This pops up the **Y Axis Format** dialog from which various y-axis properties can be set. These are described in detail in Section 28.1.5.
- 7.** Click on the **Panel 2** node in the Chart Format window to select the second panel. Then repeat steps **3** to **6** to add the desired variables to this panel and specify its y-axis properties. Repeat for as many panels as required (new panels are added using the **Add Panel** button).



Note that there are no limits to how many panels can be shown in a chart, or how many series can be displayed in a panel. In practise, however, the chart will quickly become to very crowded and difficult to interpret as panels and series are added.

- 8.** Use the **Move Up** and **Move Down** buttons to rearrange the panel order if required. First select the panel node to be moved (eg. **Panel 1**), then click on the appropriate button to move it up or down in the panel order. Note that panels are numbered so that **Panel 1** is closest to the top of the graph.
- 9.** Click on the **General Options** button to adjust chart properties such as whether a legend is required and to specify a title that is different from the default (which is generated from the **Run Identifier** string).
- 10.** Optionally save the format by clicking on the **Save Format** button. In the **Save Chart Format** dialog, specify a name for the saved format, then click **Ok**.
- 11.** Click **Ok** to exit the **Chart Specification** dialog and draw the chart.

Note that the **Remove** button can be used at any time to remove the currently highlighted node. For example, if Panel 2 is highlighted, clicking on the **Remove** button will remove it and any axes and data series contained within that panel. Similarly, the **Edit**, **Move Up** and **Move Down** buttons also operate on the currently selected node.

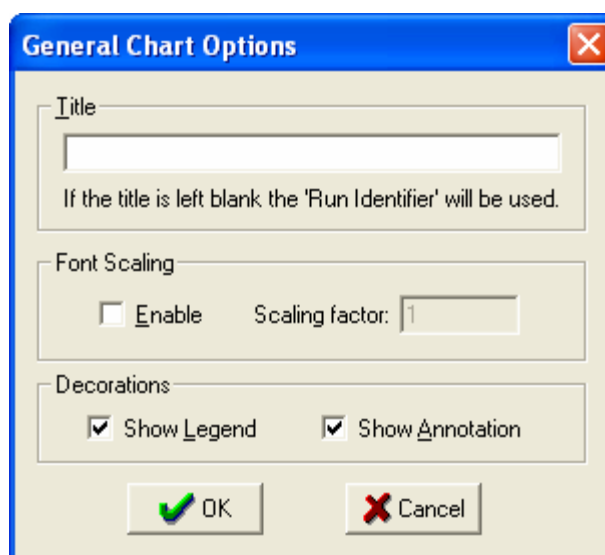
The **Save Format** button is used to save the chart format (in the *Simulation File*) so that it can be used for later charts either in the current or a future modelling session. When this button is clicked, a small dialog shows all of the currently saved formats and allows the format set in the dialog to be given a name for future reference. This is highly recommended for all but the simplest charts, as creating a chart from scratch can be a time-consuming procedure (and in practice, the same charts formats tend to be used repeatedly when running simulations). It is important when choosing format names to make sure they describe what they contain (names such as "*Beetle Development Times*" are preferable to "*dev*").

## 28.12 Chart Properties

Selecting **General Options** from the **Chart Specification** dialog allows the user to set some properties of the complete chart (Fig. 28-5).

- **Title** – this sets the title, which is plotted above the chart, centred in the middle of the window or page. If left blank, the Run Identifier (see Section 25.3) is used as title.
- **Font Scaling** – sets the font scaling used for this chart as a relative value from the default. For example, a value of 1.5 will make all text in the chart 50% larger than it would otherwise be. Note that this scaling is cumulative with the scaling set in the **Operating Preferences** dialog (Section 25.1).
- **Decorations** – specifies whether the legend and/or annotations are to be shown for this chart.

Fig. 28-5 Specifying Chart properties.



Note that these chart options refer to the whole chart. It is possible to include more than one “sub-Chart” in the complete chart by using the **Add Chart** button in the **Chart Specification** dialog (Fig. 28-2). A second chart would be shown in that dialog as **Chart 2**. Properties of the sub-charts are changed by clicking on the sub-chart node (eg, **Chart 2**) and then clicking the **Edit** button. Currently, only a title can be specified for the sub-charts.

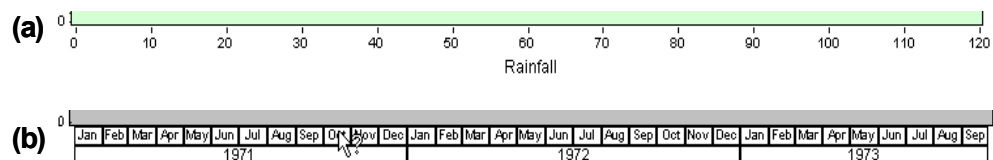
## 28.13 X-Axis Properties

The X Axis is always drawn horizontally along the bottom of the lowest panel of a chart. X-axis properties are set by selecting the **X Axis** node in the **Chart Format** section of the **Chart Specification** dialog and then clicking on the **Edit** button. This opens the X Axis Format dialog (Fig. 28-7). This relatively complicated dialog contains two separate sections, only one of which will be available at any one time, depending on which axis “style” is selected. The styles are:



- **Ticks** – this will create a standard axis, consisting of a line with tick marks and tick mark labels (Fig. 28-6a).
- **Boxes** – this axis can only be used when the x-axis variable represents dates (for example, Simulation Date). The axis consists of a set of boxes, each representing a month or a year (Fig. 28-6b).

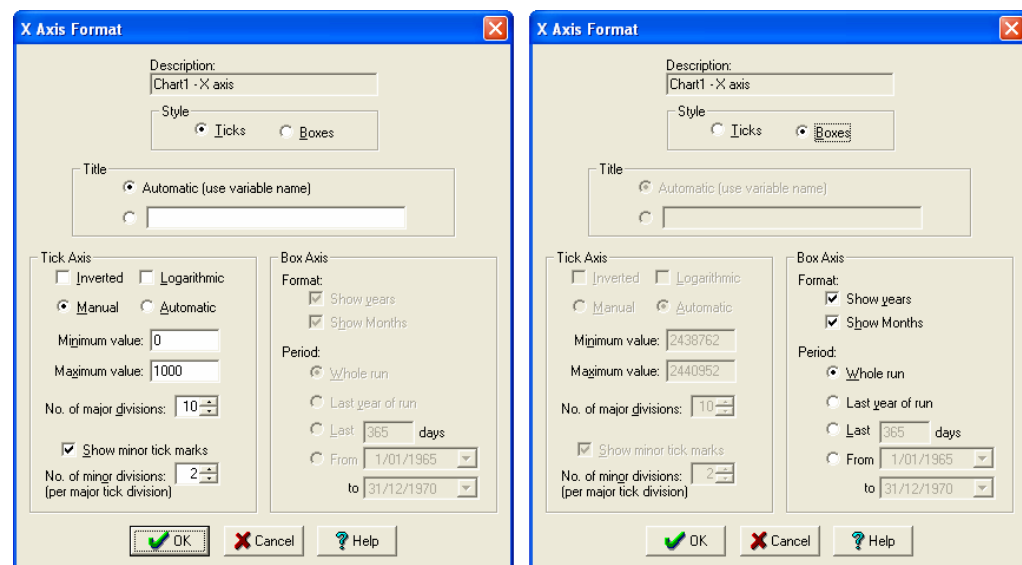
**Fig. 28-6 (a) “Tick” style X-Axis showing major divisions but no minor divisions, and (b) “Box” style X-Axis.**



### Tick Axis properties

- **Title** - A tick axis has a title, which can be **Automatic** (i.e., it will use the name of the x variable), or a name can be explicitly specified.
- **Inverted** – Normally, the smallest value on the x axis is at the far left of the axis. However, if this box is checked, the axis will be reversed, with the smallest value at the far right.
- **Logarithmic** – check this box if the axis is to have a logarithmic rather than a linear scale. Such a scale would be appropriate if there is large range (spanning many decades) in the value of the variable.

**Fig. 28-7 X Axis Format dialog (a) using normal “Tick” axis style and (b) using a “Box” axis style.**



- **Manual/Automatic** – If the Automatic option is chosen, DYMEX will choose a minimum and maximum value for the axis that fits the range of the x variable (while trying to maintain relatively “nice” values for the tick



points). If the Manual option is selected, then the user is responsible for providing the minimum and maximum values of the axis.

- **Minimum value/Maximum value** – these two values define the range of the axis when the Manual option is chosen.
- **No. of major divisions** – The number of divisions that the axis is to be divided into by the major tick marks.
- **Show minor tick marks** – if this option is checked, then minor (somewhat shorter) tick marks will be used to divide the intervals between the major tick marks.
- **No. of minor divisions** – select the number of divisions between major tick marks here if the previous option is checked.

#### **Box Axis properties**

- **Show Years** – Shows a line of boxes along the axis that each represent one year. Each box will be labelled with the year number (eg. 1986). Note that if the *Simulation Date* variable is not used for the x-variable, this option is unavailable.
- **Show Months** – Shows a line of boxes along the axis that each represent a month. Each box will be labelled with the month name (in full, where there is sufficient space, or shortened to 3 or 1 letters). If Year boxes are also present, the month boxes are placed above the year boxes.
- **X axis extent** – Use the “radio” buttons to select the full range of the run for the x axis (**Whole run**), or to show only the **last year of run**. Alternatively, a number of days to show can be specified (always taken from the end of the run) or a date range can be chosen. The latter option is unavailable (greyed out) if the *Simulation Date* is not used as x-variable.

Note that the box axis does not have a title.

### **28.14 Panel Properties**

Only a single property (the background colour) is currently settable for the panels. This is set by selecting the appropriate panel’s node in the **Chart Specification** dialog and then clicking on the **Edit** button to obtain the **Panel Format** dialog.

### **28.15 Y-Axis Properties**

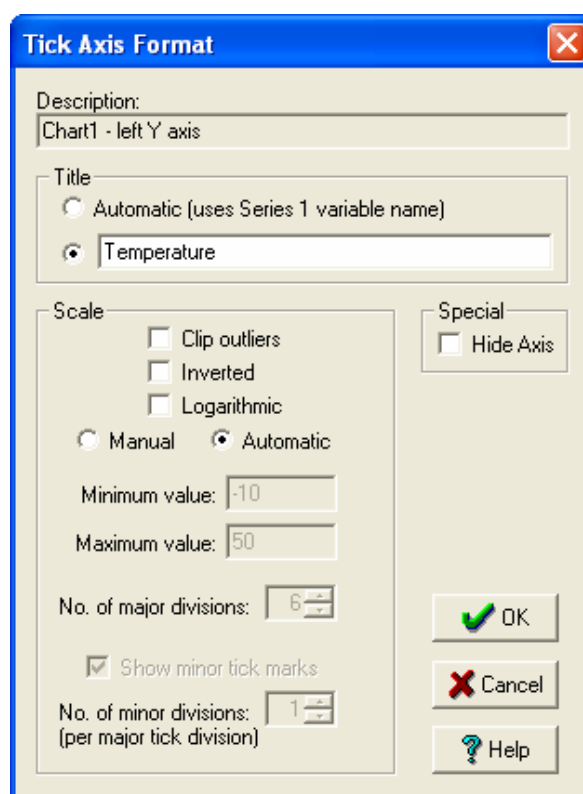
Y Axes are always drawn along the vertical edges of the panel that they belong to. Each panel can have up to two y-axes, one along the left side and one along the right side of the panel. Each series in the panel can then be associated with one or the other of these axes. By default, only a single (left) axis is created initially and all series added to the panel use this axis. To create a second

(right) axis, choose the **Use right axis** option when adding a new series to the panel (see below).

Y-axis properties are set by selecting the appropriate **Y Axis** node in the **Chart Format** section of the **Chart Specification** dialog and then clicking on the **Edit** button. This opens the Y Axis Format dialog (Fig. 28-8). The axis properties are then set as follows:

- **Title** - A tick axis has a title, which can be **Automatic** (i.e., it will use the name of the variable used as the first series in the panel), or a name can be explicitly specified. If more than one series is used, it will generally be preferable to choose a name explicitly.
- **Clip outliers** – if this box is checked, any points (or parts of lines) that fall outside the axis boundaries and belong to series that use this axis will not be drawn. In other words, the series graph is kept within the panel boundaries and points outside those boundaries are not drawn.

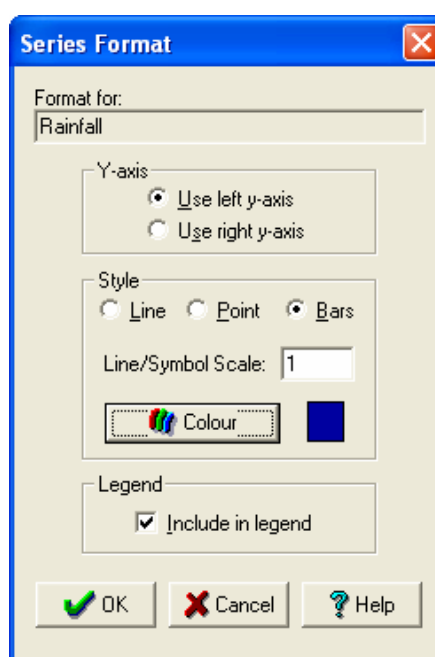
**Fig. 28-8 Y-Axis Format dialog.**



- **Inverted** – Normally, the smallest value on the y axis is at the lower end of the axis. However, if this box is checked, the axis will be reversed, with the smallest value at the top.
- **Logarithmic** – check this box if the axis is to have a logarithmic rather than a linear scale. Such a scale would be appropriate if there is large range (spanning many decades) in the value of the variable.

- **Manual/Automatic** – If the Automatic option is chosen, DYMEX will choose a minimum and maximum value for the axis that fits the range of the series variables (while trying to maintain relatively “nice” values for the tick points). If the Manual option is selected, then the user is responsible for providing the minimum and maximum values of the axis.
- **Minimum value/Maximum value** – these two values define the range of the axis when the Manual option is chosen.
- **No. of major divisions** – The number of divisions that the axis is to be divided into by the major tick marks.
- **Show minor tick marks** – if this option is checked, then minor (somewhat shorter) tick marks will be used to divide the intervals between the major tick marks.
- **No. of minor divisions** – select the number of divisions between major tick marks here if the previous option is checked.

**Fig. 28-9 The Series Format dialog.**



## 28.1.6 Series Properties

A series is the graph (either a set of markers, a line connecting a set of points or a set of bars) corresponding to one output variable. The properties of a series can be set when it is created or they can be edited later. Click on the node representing the series in the Chart Specification dialog and then click on the Edit button to obtain the Series Format dialog (Fig. 28-9). The series properties are then set, as follows:

- **Y-axis** – Indicate which y-axis to use for scaling the series within the panel.
- **Style** – Several properties can be set under this heading:
  - **Line** – plots the values of the series variables as points connected by straight lines
  - **Points** - plots the values of the series variables as marker symbols centred on each of the data points. The symbol used is a filled circle (●) and this is not changeable.
  - **Bars** – plots each data value as a vertical bar extending from the x-axis to the data point
  - **Line/Symbol Scale** – the thickness of the lines and bars and size of the marker symbols can be adjusted from the default setting (1.0) by typing a different value into the edit box. For example, a value of 2 for a line series would double the line thickness.
  - **Colour** – the colour of the line, marker or bar can be changed by clicking on this button and choosing from a colour palette. The current selection is shown in the rectangle next to the button.
- **Legend** – check the box if this series is to be included in the chart legend. Note that this choice will only take effect if a legend has actually been selected for the chart in the Chart Properties (Section 28.1.2).

### 28.1.7 The Chart Window and its Menu

The chart is displayed in a Chart Window (see Fig. 28-3). If the whole chart is composed of more than one “sub-chart” (see Section 28.1.2), DYMEX arranges the sub-charts according to a fixed algorithm (over which the user has no control). Any legends are also automatically placed when the chart is first drawn. These, however, can be moved anywhere by “dragging” them to the required position. Note that one of a number of cursor shapes is used to indicate the types of operations that are possible at any time in the window. These are detailed in the following explanation of chart operations. All of these operations can be accessed from the Chart menu, while some are also available from the tool bar.

- **Query Mode** – places the Chart Window into the “Query” mode. This mode is indicated by a cursor with a pointer and question mark (Ⓜ?). When the window is in this mode, and the cursor is placed at one of the series data points, the values of the x and y variables represented by that point are shown in the status bar, as shown below:

Simulation Date = 4/12/1971, Rainfall = 29

- **Pan Mode** – places the Chart Window into the “pan” mode and is indicated by the pan cursor (Ⓜ). In this mode, with the mouse button

down, the chart can be moved around the Chart Window. When the mouse button is released, the chart will stay where at its last position. Note that unless the mouse button is clicked directly over the legend, the legend is moved along with the chart. If the mouse is clicked over a legend, only that legend is moved.

- **Zoom Mode** – places the Chart Window into “zoom” mode, indicated by the zoom cursor (🔍). This then allows more detailed examination of any region of the chart. Two methods can be used to zoom in onto a point or region. Left-clicking on a chart point in zoom mode causes an enlarged version of the chart to be drawn (the enlargement is 50%), positioned so that the point that was clicked stays in the same absolute position on the screen. Similarly, right-clicking on a point draws a 50% smaller version of the chart, again positioned so that the clicked point does not move. Note that there are limits to the enlargement and reduction allowed for a chart. An alternative way to zoom into an area is to left-click on the point that is required as the top-left corner of the zoomed area and, leaving the mouse button down, move the mouse to the desired lower-right corner of the area. A rectangle outlining the zoomed area will appear as the mouse is moved on the screen. When the mouse button is released, the area outlined by the zoom-rectangle will be drawn to occupy the full size of the Chart Window.
- **Show all** – without changing the display mode, redraws the chart at its original size and placement in the Chart Window (i.e., it removes all zoom and pan actions applied since the chart was originally created).
- **Export Metafile** – exports the chart to a Windows Metafile (a vector format with file extension .emf), which can be imported and edited in other applications such as Microsoft Word and Powerpoint. The user is prompted for a filename before the file is written. Since the metafile format is a vector format, individual elements such as lines and labels can be accessed and edited separately using (say) Powerpoint. You will need to “ungroup” the picture elements before editing individual components.
- **Export GIF** – exports the chart as a bitmap type file in GIF format), which can be imported into other applications such as Microsoft Word and Powerpoint. Note that this format is not suitable for detailed editing as individual elements of the chart are not represented as separate drawing objects in the file.
- **Edit Format** – displays the **Chart Specification** dialog so that properties of the chart can be changed.
- **Save Format** - saves the current chart’s format (this is the same as clicking on the **Save Format** button in the **Chart Specification** dialog).
- **Delete Format** – shows a dialog that lists the entire set of chart formats created for the model and allows user to select formats for deletion.

## 28.1.8 Printing Charts



When a chart is displayed on the screen, it is possible to print it using the File menu's **Print** or **Print Preview** options. Before printing the chart, go to **Print Setup** (also on the **File** menu) and change the settings to your preferences. The chart is printed with the same aspect ratio as it is displayed on the screen, thus selecting **landscape** orientation will allow the chart to better fill the page. In the printed chart, various dashed and dotted lines replace the coloured lines. The Zoom button in the Print Preview can be used to examine the appearance of the graph in some detail.

## 28.2 Creating Tables


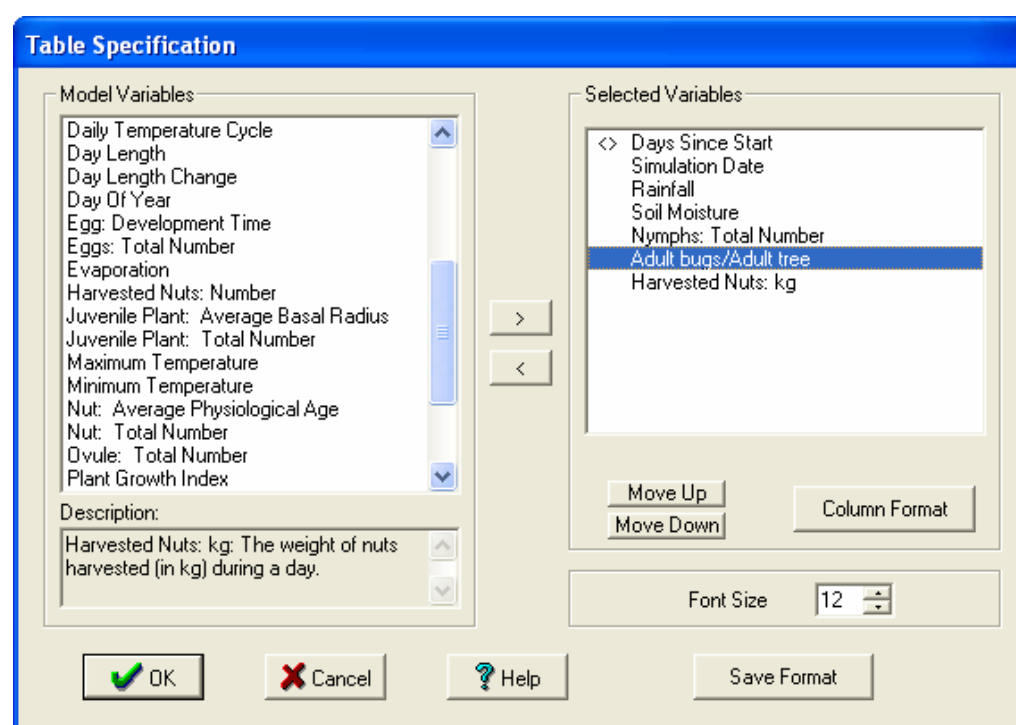

Tables are simple to create, save and export in DYMEX. Once a simulation is complete either select the table button () from the toolbar or select **Table** from the **Results** menu. If one or more table formats have been saved from earlier runs, a popup menu appears, allowing a choice of format. Choose **New** to create a new table format, or select the format required. The **Table Specification** dialog (Fig. 28-10) can then be used to specify which variables to include in the table. Note that this dialog is slightly different for "Single" and "Multiple" runs.


Fig. 28-10 The Table Specification dialog box used for "Single" runs.



### ➤ To add variables to the table

1. Select the variables from the **Model Variables** list by clicking on them. More than one variable at a time can be selected. Selected variables can be un-selected by clicking on them again.

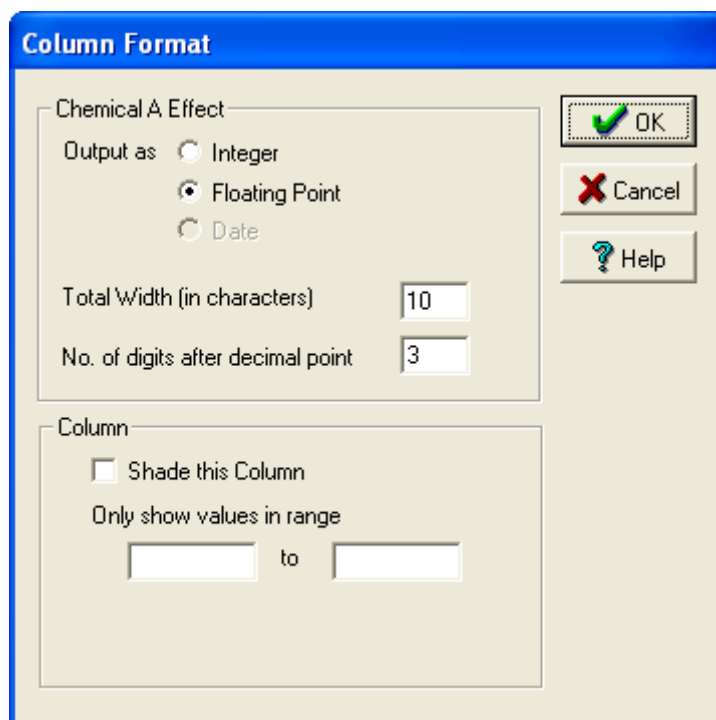
2. Either double-click them or left click the “Add” button (  ) to add them to the **Selected Variables** list at the right.

The selected variables are shown in the **Selected Variables** list. A variable that has been added by mistake can be removed from the **Selected Variables** list by selecting it and then clicking on the “Remove” button (  ). The contents of the columns in the table will appear in the order that the variables appear in the **Selected Variables** list. The **Move Up** and **Move Down** buttons can be used to change the order of the variables in the list.

The format in which a variable is to be presented within the table can be changed from the default by selecting the variable and left clicking the **Column Format** button. This displays the **Column Format** dialog box (Fig. 28-11), which allows the variable to be formatted for the table. The variable can be presented as an integer or a floating-point value with a user-defined number of decimal places. If the data in the column is a date (for example, *Simulation Date*), the date format to be used can be chosen from a list. The width of the column can be specified (as a multiple of the width of an average character in the font being used) in the **Total Width** edit field. Note that if the Total Width is set to 0, the column width will be automatically adjusted to fit the data being displayed.

Tables from “*Multiple*” runs will show an extra option in the **Column** panel, that allows the table rows to be sorted in either ascending or descending order using the specified column as the sort key. More than one column may be specified to act as sort key, in which case the leftmost column acts as the primary sort key, and so on.

**Fig. 28-11** The Column Format dialog as it appears for numeric data in tables for “Single” runs.



The Column panel specifies some additional treatments for the data display in the selected column. Selecting **Shade this Column** applies a background colour to the column to highlight it. The two edit boxes under **Only show values in range** are used to set optional minimum and maximum value of data to display in the table. For example, values of 366 and 730, respectively, for the *Days since Start* variable would create a table containing the output from the second year of the simulation only. The *Days since Start* variable has been restricted in such a way in Fig. 28-10, and this is indicated by the “◁” indicators in front of it in the **Selected Variables** list.

A font size can be selected for the table display in the **Font Size** panel. Smaller font sizes can be useful in fitting all columns of a table across a printed page.

Once the format for the column and table has been chosen then the format of the table can be saved that will allow its reuse for other simulations. To save the format left click the **Save Format** button and choose a name for the format. Once a format has been saved, the format name will appear in the popup menu the next time a table is required.

**Fig. 28-12 An example table produced by a “Single” run, with one column shaded.**

Table - Met Data and Bugs [Run 2]							
SimulaDate	T-min	T-max	Rain	SoilMoistU	Bugs/tre	gg:DeveTir	uPIAwBaR
Run 2 (14:35) Tough Nut Orchard model; ROC.DAT (1 Jan 71)							
1/1/1971	23.00	38.00	0	0.4	0.11	5.00	0.1
2/1/1971	23.00	41.00	0	0.3	0.21	5.00	0.1
3/1/1971	26.00	41.00	0	0.3	0.32	5.00	0.1
4/1/1971	25.00	37.00	0	0.3	0.42	6.00	0.1
5/1/1971	24.00	34.00	0	0.3	0.53	6.00	0.1
6/1/1971	24.00	31.00	0	0.3	0.63	6.00	0.1
7/1/1971	23.00	32.00	0	0.3	0.74	6.00	0.1
8/1/1971	21.00	34.00	0	0.3	0.84	6.00	0.1
9/1/1971	21.00	34.00	0	0.3	0.95	6.00	0.1
10/1/1971	19.00	34.00	0	0.2	1.05	6.00	0.1
11/1/1971	21.00	34.00	0	0.2	1.16	6.00	0.1
12/1/1971	21.00	34.00	0	0.2	1.26	6.00	0.1
13/1/1971	22.00	33.00	0	0.2	1.37	6.00	0.1
14/1/1971	21.00	31.00	29	0.5	1.47	6.00	0.1
15/1/1971	22.00	33.00	0	0.5	1.58	6.00	0.1

The shaded section of the table across the top shows the Run Identifier (see Section 25.3) for the run that produced the table.

The **Table Specification** dialog (Fig. 28-10) is slightly different when used on the results of a “*Multiple*” run. An additional list (**Row Labels**) at the top



right-hand side of the dialog shows the names of additional columns that are available to label each row of results (Fig. 28-13). A ‘\*’ indicator at the left marks those row labels that have been selected for inclusion in the table. To select or deselect a label, select the label and then click the “+/-” button. Note that the row labels will vary with the type of Sequence being run. Row label columns are always placed at the left of the table, and have a blue background. Fig. 28-13a shows the row labels available with a MetManager Sequence. The labels consist of the names of the Continent, Country, State and Location represented by the row. Fig. 28-13b is a typical set of row labels from an Event Sequence. In that Sequence, the row labels indicate the time of an event trigger. Note that ‘Vac’ in this case is the event Sequence’s Event Type Label (see Section 27.1, *Event Run Sequences*), with the following number being the index of the application. Hence, if the Event module applied vaccinations, the ‘Vac2’ row label would be the day of application of the 2<sup>nd</sup> vaccination. Fig. 28-13 shows how each of these row label specifications will be translated into a set of row labels in the resulting table.

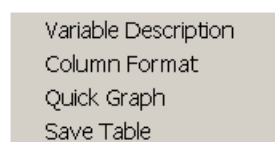
**Fig. 28-13 The Row Labels selector and resulting row labels for (a) a MetManager controlled sequence, and (b) and Event module controlled sequence.**

Continent	Country	Location	GI [1]
Run (12:17) Australia (Level 5); Diuraphis noxia (Russ)			
Oceania	Australia	Adelong	30
Oceania	Australia	Albury	31
Oceania	Australia	Armidale	31
Oceania	Australia	Balranald	61

Vac1	Vac2	Vac3	Total (\$)
Vaccination (3); Animal Breed=European			
1	91	181	188.14
8	98	188	198.86
15	105	195	201.92
22	112	202	200.00

With a table displayed, double clicking a column of data (with the exception of row labels) will cause the menu shown in Fig. 28-14 to appear. Several actions are available from this menu, and these are described below:

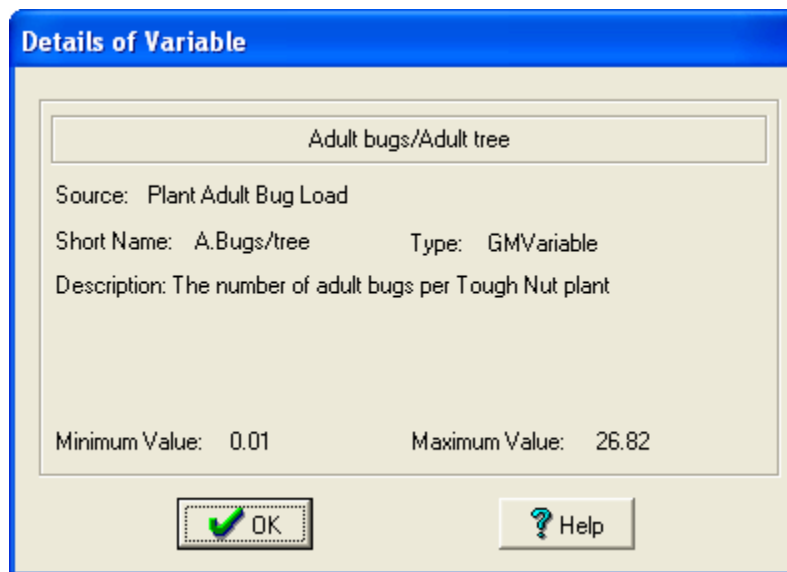
**Fig. 28-14 The menu from double clicking the table window.**



- **Variable Description** – shows a summarised description of the variable whose values are displayed in the clicked column. Fig. 28-15

shows the results of a **Variable Description** request for the variable “*Adult bugs/Adult tree*” (the shaded column in Fig. 28-12). The **Variable Description** dialog box shows the full name of the variable in the panel at the top. Also shown are the name of the “source” module, the short name (mnemonic) of the variable, which is used as the column header, the **Maximum** and **Minimum Values** of the variable over the simulation period, the **Type** of variable it is and its **Description**. Note that a description is only available if the model designer has provided one when building the model.

**Fig. 28-15** An example **Variable Description** dialog box for “*Adult bugs/Adult tree*”.

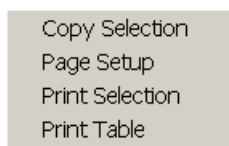


- **Quick Graph** - A chart of the variable plotted against time can be created by double clicking the column and choosing **Quick Graph**. This will display a graph of the variable plotted against time. DYMEX chooses a default format for this graph, which is used for all Quick Graph charts. Note this format is saved in the Simulation File with the name “<**QuickGraph**>”, along with all other chart formats and may be edited in the same way (see 28.1.1).
- **Column Format** – shows the **Column Format** dialog (Fig. 28-11) to allow changes to be made to the way the data in the selected column is displayed.
- **Save Table** - saves whole table as a comma-delimited file (\*.csv) which can be easily imported into an Excel spreadsheet. When the **Save Table** button is clicked a file dialog box will appear requesting a file name and directory.

Right-clicking By highlighting an area of the table and right clicking the mouse over it the menu shown in Fig. 28-16 will appear, giving a range of options that apply mainly to printing or copying the table or a selected portion of it. A portion of the table can be selected by clicking on the top left cell of the required region

and, with the mouse button down, moving the cursor to the bottom right cell of the region. The selected region will be highlighted in a dark colour.

**Fig. 28-16 The menu from right-clicking the table window.**



**Copy Selection** - the selected section of the table will be copied to the clipboard, from where it can be pasted to other MS Windows programs.

**Print Selection** **Print Selection** allows the printing of any section of the table. When **Print Selection** is selected the **Print** dialog will be displayed.

**Printout Scaling** **Page Setup** makes available many options for setting up the appearance of a printed Table page. Footers and Headers can be added to the page, grid lines turned on or off, and margins set. A particularly useful feature in this dialog is the Scale option, where the size of the table can be scaled down and up, thus allowing relatively large tables to be printed in compact form.

**Print Table** **Print Table** prints the entire table, and is the same as selecting the **Print** option from the **File** menu.

## 28.3 Creating Maps

Maps can be created after a “**Multiple**” run, if the individual simulations in the Sequence included different locations, and the latitude and longitude of each location is available in a Summary Variable. This generally means that maps are available only for runs controlled by either a **DataFile** or **MetManager** Sequence. For all other types of simulation runs (including all “**Single**” runs), the **Map** menu item and the map button (🗺️) on the toolbar will be disabled. Several different types of map display are available. If the run consists of a number of discrete locations (perhaps representing towns or cities), output variables can be plotted as symbols of differing sizes or colours (with the size or colour dependent on the value of the variable) or as a number representing the variable directly. Once a map has been created, DYMEX provides a way to save the format of that map, allowing quick loading of maps with the same format later. Maps can be copied to word processors and printed in the same way as charts.

All DYMEX maps consist of two major components:

1. A **Region Definition** Layer defines the extent of the map that is shown and the geographical features included. Region Definitions are stored in Map Region Definition files (.MDF) in the map directory. These

files are created and modified using the MapManager, which is detailed in Section 29.

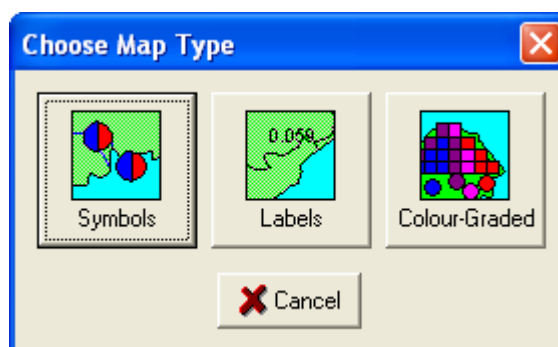
2. The **Data** Layer is the representation of the DYMEX output variables plotted on the map. It specifies which variables will be shown on the map and in what way that variable will be represented.

Note that you cannot draw any maps unless at least one Region Definition file is present. DYMEX includes a set of these files (one covering the world, and one for each continent) and these can be used for creating maps initially. These files can later be edited or new ones created if the supplied maps are not adequate for a particular purpose.

➤ **To create a map**

1. Select **Results** from the main menu and choose **Map** or left click the map button on the tool bar. If any map formats have been saved, this will result in a small popup menu, giving a choice of map formats. Select the **New** option from this menu to create a new map type, or select one of the saved map formats to create a map with the same format, but using the data from the current run.

**Fig. 28-17 The Map Type dialog.**



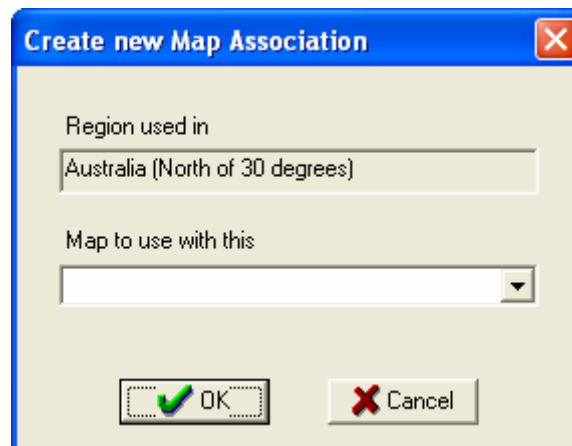
2. If you have chosen a **New** map, the **Choose Map Type** dialog will appear (Fig. 28-17), giving a choice of the data is to be plotted on the map. Click on the appropriate button to choose either “**Symbols**” (Section 28.3.1), “**Labels**” or “**Colour-Graded**”. This will display the **Map Specification** dialog for the chosen map type. See the sections on the appropriate map types for more information (Section 28.3.1, *Symbols Map Specification*, Section 28.3.2, *Label Map Specification*, or Section 28.3.3, *Grid (Colour-Graded) Map Specification*).
3. If you have chosen a saved map format, but the Sequence used for the run has not been mapped before, the **Map Association** dialog (Fig. 28-18) will appear.

DYMEX cannot know in advance which *Map Region Definition* is most appropriate for a particular set of locations (the Sequence) and the **Map Association** dialog allows the user to specify this association. In later runs,

when the same Sequence is run, the associated *Map Region Definition* will then automatically be used in the map. Map Associations are stored in the Simulation File and are automatically used in future when the model is run with the same Sequence and a pre-defined map format is chosen for display. The region associated with the Sequence is substituted for the region defined in the map format before the map is drawn. Map Associations can be deleted, if required, using the **Delete Associations** menu item in the **Map** menu (see Section 28.3.4).

4. Choose a *Map Region Definition* by clicking on the small button to the right of the box marked **Map to use with this Region**, and selecting one of the regions from the drop-down list.
5. Close the **Map Association** dialog by clicking **Ok** and the required map will be drawn.

**Fig. 28-18 The Map Association dialog.**

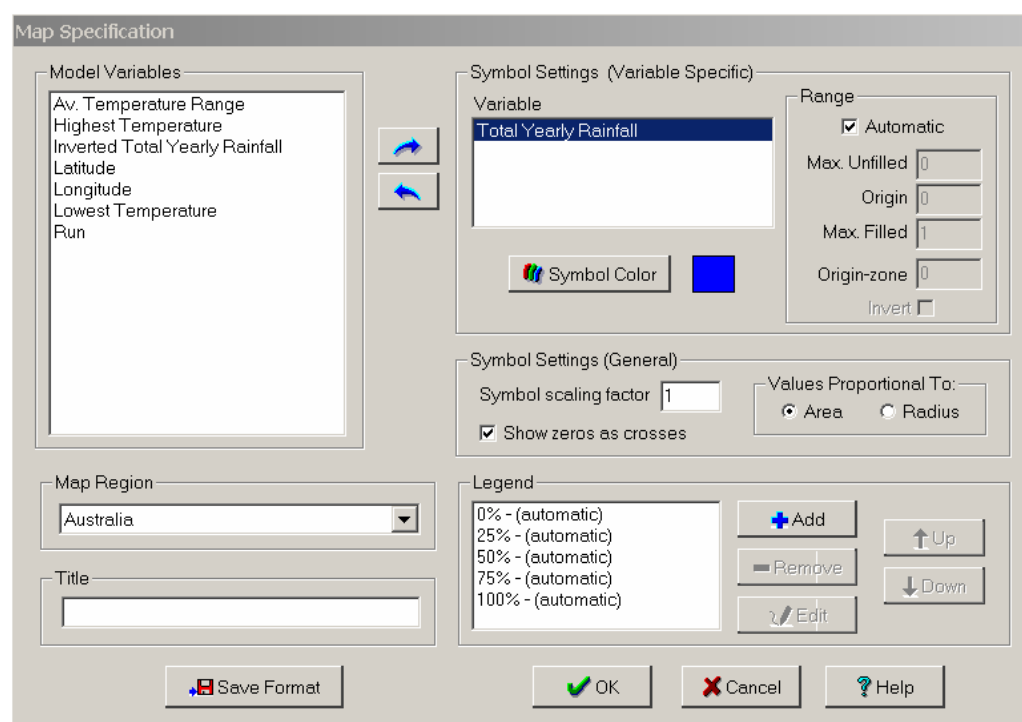


Note that if you select a saved format, the map is drawn without the Map Setup dialog being shown. This behaviour can be changed by checking the “*Always show Display Dialog*” option in Operating Preferences dialog (reached via the **Preferences|Operating** menu item).



### **28.3.1 Symbols Map Specification**

A Symbols map plots the data as a circle, the size of which is proportional to the value of the data being represented. More than one variable can be plotted on the same map, in which case the variables will be shown as different coloured circle segments. The Symbol Map Specification dialog (Fig. 28-19) is used to specify various features of a symbol map.

Fig. 28-19 The Map Specification dialog for “Symbol” maps.

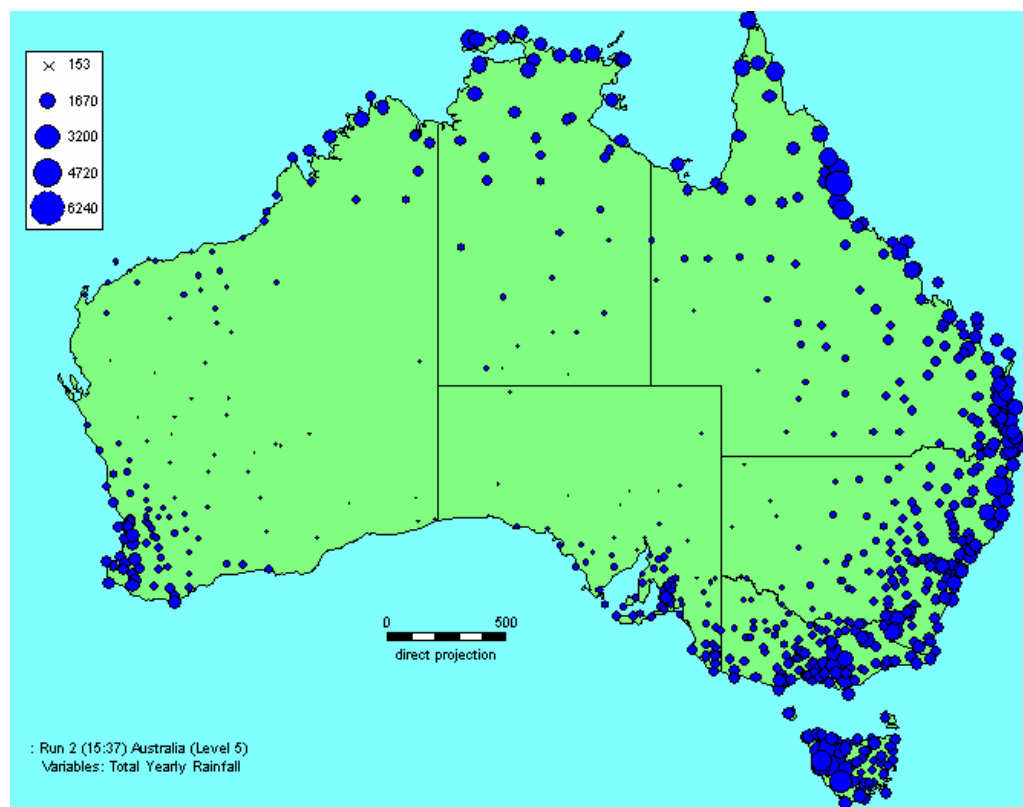


➤ **To create a simple map (with automatic scaling and legend)**

1. Highlight a variable to be plotted on the map in the **Model Variables** list by clicking on it and then click on the add button (  ). This adds the variable to the **Variable** list, which contains the names of the variables to be plotted. A variable that is added by mistake can be removed by selecting it in the Variable list and clicking on the remove button (  ).
2. Select a *Region Definition* from the **Map Region** dropdown list. If no region is displayed here, either none have yet been defined or the map directory is not correctly set (see Section 25.1, *Operating Preferences*).
3. Click Ok to draw this map with all other settings at their default values.

A map drawn using these settings is shown in Fig. 28-20. Note that a title has been automatically generated from the model’s run identifier, the circles have been automatically scaled between the variables minimum and maximum values and a legend has been generated. Whether a title, legend or scale is shown on the map, as well as the initial position of these items, is specified in the **Map Region Definition**. The Map Region Definition specifies such features as the map’s background colour and the colour of its geographical features. Often, the default settings for a map are all that are required. However, the Map Specification dialog offers many options for changing the way that the data is displayed on the map. These options are now explained in detail.

**Fig. 28-20** An example of a DYMEX “Symbol” map plotting the annual rainfall in Australia.



The **Model Variables** listbox shows all the output variables of the model that are suitable for plotting on a map. This includes all the Summary Variables except for the latitude and longitude variables. The variable or variables to be plotted on the map are moved into the **Variable** listbox, as outlined above. Various plotting options are then available for that variable in the **Symbol Settings (Variable Specific)** panel. If more than one variable has been placed into the **Variable** list, make sure that the appropriate variable is first selected (highlighted) before changing the options.

- **Symbol Colour** – this button allows the colour of the circle to be selected. The currently selected colour is shown to the right of the button.

The **Range** panel contains options for how each circle is to be scaled and represented. Note again that this applies only to the currently selected variable.

- **Automatic** – scales the symbols so that the circles will be at a minimum size (a dot) for the lowest value of the variable and at the maximum size for the highest value of the variable. The **Symbol Scaling Factor** (see below) can be used to adjust the maximum size. If **Automatic** is unchecked, the values of the variable corresponding to minimum and maximum circle size must be specified using the **Max**.

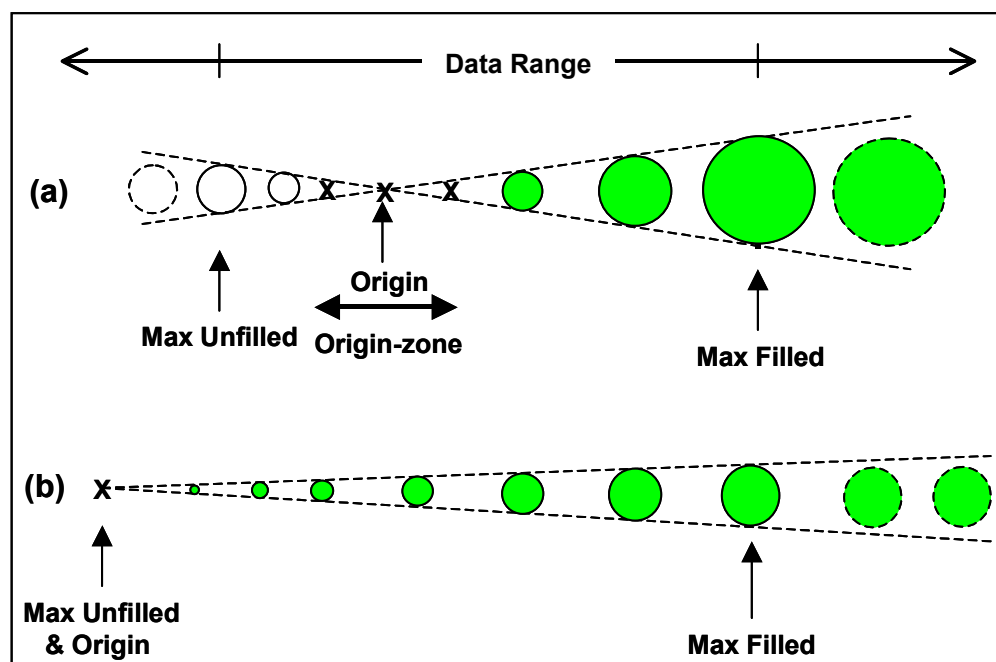
**Unfilled, Origin** and **Max. Filled** options, as below. Note that all other scaling options are disabled when **Automatic** is selected.

Manual scaling allows setting of three points within the range of possible values of the data that determine the type of circle plotted for any particular data value. Data values are converted to circles so that the largest values produce the largest **filled** circles, the smallest values produce the largest **unfilled** circles, and at a specified intermediate value (the **Origin**), the diameter of the circle is 0 (Fig. 28-21a). The circle size corresponding to the maximum filled circle is set by the **Symbol Scaling Factor**, with circle “size” grading linearly over the full range between maximum unfilled and maximum filled. From this it follows that the size of the largest **unfilled** circles depends on the ratio

$$(Max\ Filled - Origin)/(Origin - Max\ Unfilled).$$

Note that the **Automatic** option corresponds to the largest unfilled value and Origin being equal (and at the smallest data value), and the largest filled value set to the largest data value (Fig. 28-21b).

**Fig. 28-21 The relationship between data values and symbol types and sizes plotted on the map.**



- **Origin** - specifies the data value for which the diameter of the circle will be 0.
- **Max. Filled** – specifies the data value for which the size of the filled circle should be at its maximum. Data values greater than this value will be shown as maximum size filled circles with dashed borders.
- **Max Unfilled** – specifies the data value for which the size of the unfilled circle should be at its maximum. Data values greater than this



value will be shown as maximum size filled circles with dashed borders.

- **Origin-zone** – specifies a range of values around the origin for which either crosses will be drawn instead of circles if the **Show zeros as crosses** option is ticked, or no symbol at all will be drawn.
- **Invert** – reverses the scale, so that the circles will be at a minimum size (dots) for the value specified in the **Max. Filled** edit box and at a maximum size when the variable is equal to **Origin**.

**Fig. 28-22 Rainfall data plotted on maps using four different Range settings. See text for explanation.**

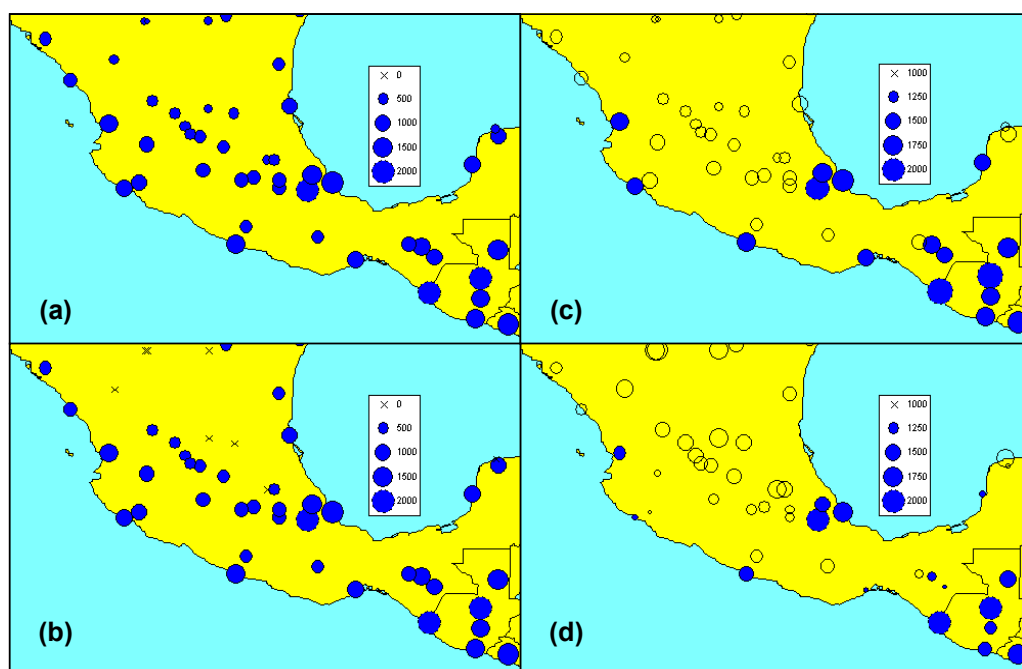


Fig. 28-22 shows rainfall data plotted on maps using different (manual) settings for the **Range**, as below:

- (a) **Origin** = 0, **Max. Unfilled** = 0, **Max. Filled** = 2000, **Origin-zone** = 0
- (b) **Origin** = 0, **Max. Unfilled** = 0, **Max. Filled** = 2000, **Origin-zone** = 500
- (c) **Origin** = 0, **Max. Unfilled** = 1000, **Max. Filled** = 2000, **Origin-zone** = 0
- (d) **Origin** = 1000, **Max. Unfilled** = 0, **Max. Filled** = 2000, **Origin-zone** = 0

Note especially the difference between maps (c) and (d). In (d), the circles are smallest for rainfalls of 1000mm (the **Origin**), with the size of the circles indicating the magnitude of the difference between the location's rainfall and 1000 mm – filled circles mark a positive difference, unfilled a negative difference.

The **Symbol Settings (General)** panel contains more options specifying how the symbols are to represent variables. These settings apply to all the variables being shown on the map.

- **Symbol Scaling Factor** – changes the scaling of all symbols on the map. The value is relative to the default symbol size (hence a value of 2 will double the size of all symbols on the map).
- **Show zeros as crosses** – this option is selected by default. If unchecked, those points on the map that would normally show as crosses (as determined by the **Range** settings) are not drawn at all.
- **Values Proportional To** – determines how the circle sizes are calculated from the variable's value. Fig. 28-21 illustrates the situation where the circle's **Radius** is proportional to the data value. However, for many situations, it may be preferable to have the **Area** of the circle proportional to the data value. Using the area tends to de-emphasise the difference between very small and very large values. In some ways, you can consider the **Area** representation as equivalent to using a log-scale for a graph.

The **Legend** panel allows the contents of the legend to be selected. To understand how the legend specification works, we need to define two values in the data range:

$P_{min}$  – the smallest of (**Origin**, **Max. Filled** and **Max. Unfilled**)

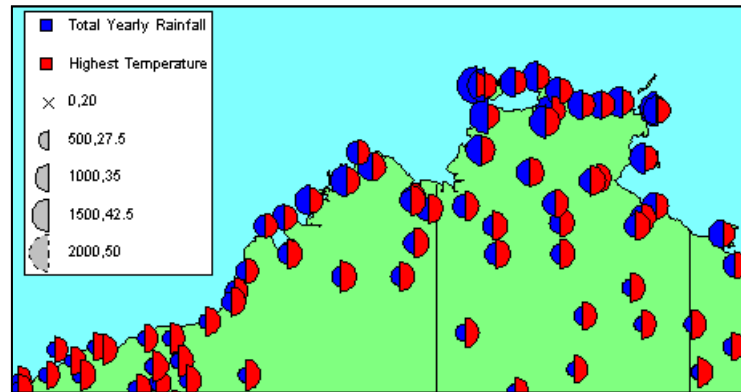
$P_{max}$  – the largest of (**Origin**, **Max. Filled** and **Max. Unfilled**)

By default, the legend contains 5 symbols, representing data at 0%, 25%, 50%, 75% and 100%, respectively, of the range between  $P_{min}$  and  $P_{max}$ . The legend item for each of these points consists of the appropriate symbol and a label identifying the value of the data corresponding to that symbol. The **Add** and **Remove** buttons may be used to add a new legend item or remove existing ones. The **Edit** button is used to change which point within the data range the legend will represent as well as the legend label. If the label field is left blank, the label will consist of the value represented by the legend. Otherwise, any text can be used to label the point. For example, a legend could consist of 3 items labelled “*Minimum*”, “*Median*” and “*Maximum*”, respectively. By default, legend items are arranged in order from minimum to maximum represented data size. The **Move Up** and **Move Down** buttons are used to change to another ordering.

The title of the map normally consists of two lines. The first is the Run Description (see Section 25.3) for the run that produced the table, while the second lists the names of the variables shown on the map. If the **Title** field at the lower-left corner of the Map Specification dialog is changed to a non-blank string, however, that string is used as the first line of the title instead of the Run Identifier.

So far we have considered only maps consisting of a single variable. If more than one variable is specified in the Variable list box, each variable is plotted as a circle segment. For example, if two variables are selected, each variable is plotted as a semicircle of the selected colour, placed so that each semicircle's common centre corresponds with the locations position on the map (Fig. 28-23).

Fig. 28-23 Two variables plotted on a single map.

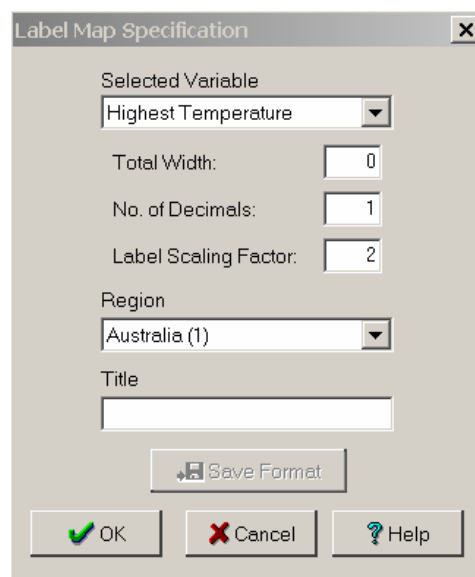


Note that the legend now contains more information. It shows the colour corresponding to each variable in the map at the top, and the values of each variable corresponding to a particular symbol size and type.

### 28.3.2 Label Map Specification

A Label map plots the data as numbers centred on the corresponding coordinates of the map. Only one variable can be plotted on the same map. The **Label Map Specification** dialog (Fig. 28-19) is used to specify various features of a Label map.

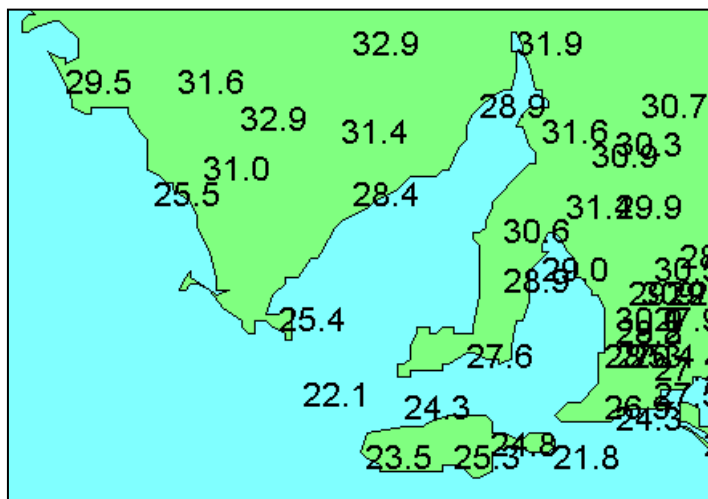
Fig. 28-24 The Label Map Specification dialog.



The variable whose values are to be plotted is selected from the drop-down list in the **Selected Variable** panel. The precision of the label is then selected by typing an integer between 0 and 10 into the **No. of Decimals** edit box. For example, if the number to be plotted is 6.14985, a precision of 0 would plot the string “6”, while a precision of 3 would plot “6.150”. The **Label Scaling**

**Factor** increases or decreases the size of the labels from the default by the factor specified. As for symbol maps, a previously created map region is selected from the **Region** drop-down list. If the **Title** field is left blank, the title consists of the Run Identifier and the name of the variable whose values are being plotted. Otherwise, the title field string replaces the Run Identifier. An example of a Label Map is shown in Fig. 28-25. Note that this map is not suitable for plotting sets of locations in close proximity. A legend is never drawn for a Label Map.

**Fig. 28-25 An example Label Map.**



### 28.3.3 Grid (Colour-Graded) Map Specification

A Grid (or Colour-graded) map plots the data as symbols (either a rectangle or circle), the colour of which depends the value of the data being represented. Only one variable can be plotted on such a map. A minimum and maximum value for plotting is specified for the variable, and colours to represent these “end”-values are chosen. DYMEX then interpolates a number of colours (as specified) between the “end”-colours, so that each colour represents a particular range of data values. Each location is then drawn in the appropriate colour. The **Colour-Graded Map Specification** dialog (Fig. 28-26) is used to specify the various options available for this style of map.

The variable to be mapped and the cut-off limits of the variable for inclusion in the map is specified **Variable** panel. The drop-down selection box labelled **Name** is used to select the variable for mapping, while the **Range** options specify the scaling, as follows:

- **Automatic** – if checked, the data range will be set so that the “minimum” colour value corresponds to the smallest value being plotted and the “maximum” colour value to the largest.
- **Show Zeros** – if unchecked, data values equal to or less than the specified **Minimum** (see below) will not be plotted on the map.

- **Minimum** – the minimum value of the data range representation. Data values at or below this value will be plotted with the selected “minimum” colour. Note, however, if **Show Zeros** is not checked, values below this value will not be plotted on the map at all.

**Fig. 28-26 The Map Specification dialog for “Colour-graded” maps.**

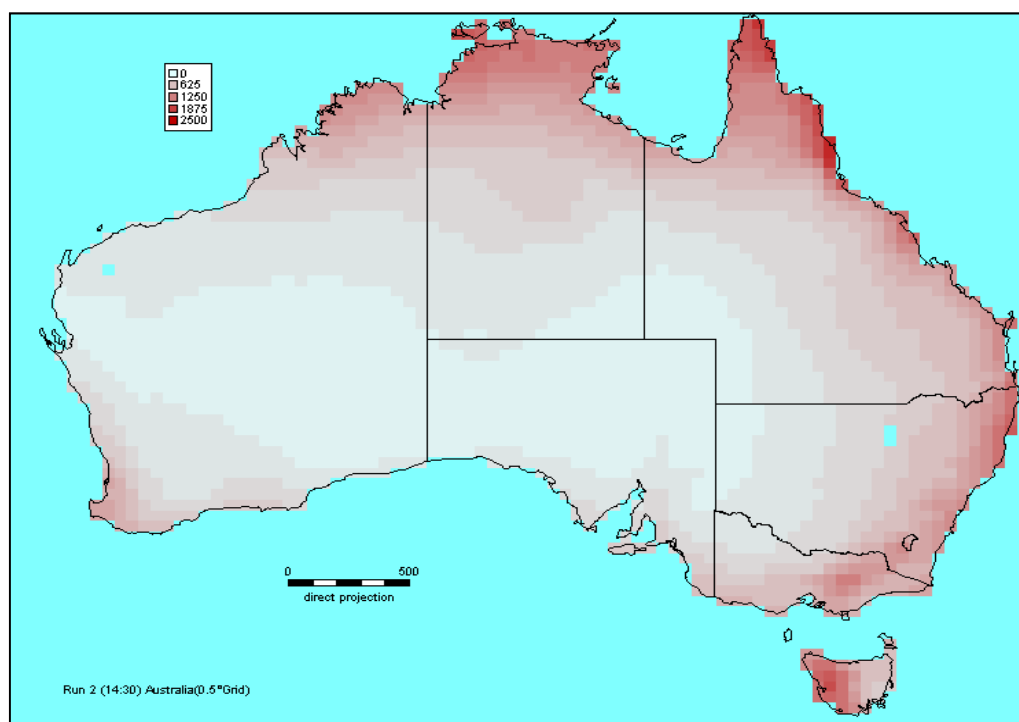
- **Maximum** – the maximum value of the data range representation. Data values at or above this value will be plotted with the selected “maximum” colour.
- **Zero Zone** – specifies a range of values around the **Minimum** for which no symbol at all will be drawn.

The **Symbol Type and Size** panel specifies the type of symbol to be drawn for each data point, as well as the size of each symbol. In this type of map, all symbols are drawn at the same size.

- **Blocks** – specifies that the symbols will be rectangles, with the given **Width** and **Height** (both in degrees). This option will be most useful for plotting locations that arranged in a regular grid pattern.
- **Circles** – each symbol is plotted as a circle with the specified **Diameter** (in degrees). This is an alternative to the “Symbol”-type map (see Section 28.3.1) that uses a colour to represent the data values rather than a different circle size.

The colours that are to be used to represent the data values are chosen in the **Colours** panel. Two colours (to represent the **Minimum** and **Maximum** values) need to be selected, as well as the number of colours that the range between minimum and maximum should be divided into. The symbols will be drawn without outlines, unless the **Show outlines** box is checked. In that case, an outline colour can be selected.

**Fig. 28-27** An example “Colour-graded” map.



As for other types of maps, a previously created map region is selected from the **Region** drop-down list. The title of the map normally consists of two lines. The first is the Run Identifier (see Section 25.3) for the run that produced the table, while the second lists the names of the variables shown on the map. If the **Title** field at the lower-left corner of the Map Specification dialog is changed to a non-blank string, however, that string is used as the first line of the title instead of the Run Identifier.

The **Legend** panel allows the contents of the legend to be selected. By default, the legend contains 5 symbols, representing data at 0%, 25%, 50%, 75% and 100%, respectively, of the range between **Minimum** and **Maximum**. The legend item for each of these points consists of the appropriately coloured symbol and a label identifying the value of the data corresponding to that symbol. The **Add** and **Remove** buttons may be used to add a new legend item or remove existing ones. The **Edit** button is used to change which point within the data range the selected legend item will represent as well as the legend label. If the label field is left blank, the label will consist of the value represented by the legend. Otherwise, any text can be used to label the point. For example, the specified **Maximum** may not be the largest value of the

mapped variable, in which case it may be preferable to change the 100% legend item from a default of (say) “**2500**” to “**>= 2500**”. By default, legend items are arranged in order from minimum to maximum represented data size. The **Move Up** and **Move Down** buttons are used to change to another ordering. Fig. 28-27 shows a “colour-graded” map and legend of rainfall in Australia.

### **28.3.4 The Map Window and its Menu**

The map is displayed in a Map Window. The Map Window shows a subset of the full world, with boundaries and colours as specified in the Map Region File. In other words, the Map Window acts as a “viewport” on a map of the world. The legend, label and scale indicator will be present if this has been specified when creating the map region file and placed where specified in the map region file. These, however, can be moved anywhere by “dragging” them to the required position, or removed if required. Note that one of a number of cursor shapes is used to indicate the types of operations that are possible at any time in the window. These are detailed in the following explanation of map operations. All of these operations can be accessed from the Map menu, while some are also available from the tool bar.

- **Query Mode** – places the Map Window into the “Query” mode. This mode is indicated by a cursor with a pointer and question mark (⏏). When the window is in this mode, and the cursor is placed anywhere on the map, the coordinates (in degrees) of the cursor point are shown at the left of the status bar.
- **Pan Mode** – places the Map Window into the “pan” mode and is indicated by the pan cursor (☞). In this mode, with the mouse button down, the map boundaries can be moved. The effect is as if a map of the world is present behind the Map Window, and the user is moving the map window over this map. When the mouse button is released, the map will stay at its last position. Note that the legend, title and scale marker stay at the same position with respect to the Map Window.
- **Zoom Mode** – places the Chart Window into “zoom” mode, indicated by the zoom cursor (🔍). This then allows more detailed examination of any region of the map. Two methods can be used to zoom in onto a point or region. Left-clicking on a map point in zoom mode causes an enlarged version of the chart to be drawn (the enlargement is 50%). The map is positioned so that the point that was clicked stays in the same absolute position on the screen (**Dymex Zoom modes**) or is centred in the zoom window (**GIS zoom mode**). See Section 25.1 for details on setting the zoom modes. Similarly, right-clicking on a point draws a 50% smaller version of the map. Note that there are limits to the enlargement and reduction allowed for a map. An alternative way to zoom into an area is to left-click on the point that is required as the top-left corner of the zoomed area and, leaving the mouse button down, move the mouse to the desired lower-right corner of the area. A rectangle outlining the zoomed area will appear as the mouse is moved on the screen. When the mouse button is

released, the area outlined by the zoom-rectangle will be drawn to occupy the full size of the Map Window. When one of the Dymex zoom modes is used, the mouse wheel can be used to zoom in or out.

- **Reset** – without changing the display mode, redraws the map with the settings specified in the Map Region File (i.e., it removes all zoom and pan actions applied since the map was originally created).
- **Show all** – without changing the display mode, shows a full map of the world (the effect is the same as zooming out until the whole world is shown).
- **Title** – toggles the title display between showing a title and not showing a title. The menu item is ticked if a title is being shown.
- **Legend** – toggles the legend display between showing a legend and not showing a legend. The menu item is ticked if a legend is being shown.
- **Scale** – toggles the scale display between showing a scale marker and not showing a scale marker. The menu item is ticked if a scale marker is being shown.
- **Projection** – allows the user to choose a different projection for display of the map. See Map Manager for details on projections.
- **Projection Parameters** – allows the user to choose different projection parameters for the current map. See Map Manager for details on projections.
- **Export Metafile** – exports the map to a Windows Metafile (a vector format with file extension .emf), which can be imported and edited in other applications such as Microsoft Word and Powerpoint. The user is prompted for a filename before the file is written. Since the metafile format is a vector format, individual elements such as lines and labels can be accessed and edited separately using (say) Powerpoint. You will need to “ungroup” the picture elements before editing individual components.
- **Export GIF** – exports the map as a bitmap type file in GIF format), which can be imported into other applications such as Microsoft Word and Powerpoint. Note that this format is not suitable for detailed editing as individual elements of the map are not represented as separate drawing objects in the file.
- **Edit Format** – displays the appropriate **Map Specification** dialog so that properties of the map can be changed.
- **Save Format** - saves the current map’s format (this is the same as clicking on the **Save Format** button in the **Map Specification** dialog).
- **Delete Format** – shows a dialog that lists the entire set of map formats created for the model and allows user to select formats for deletion.



- **Delete Associations** – enables removal of any existing *Map Associations* for the current model.

### 28.3.5 Printing Maps

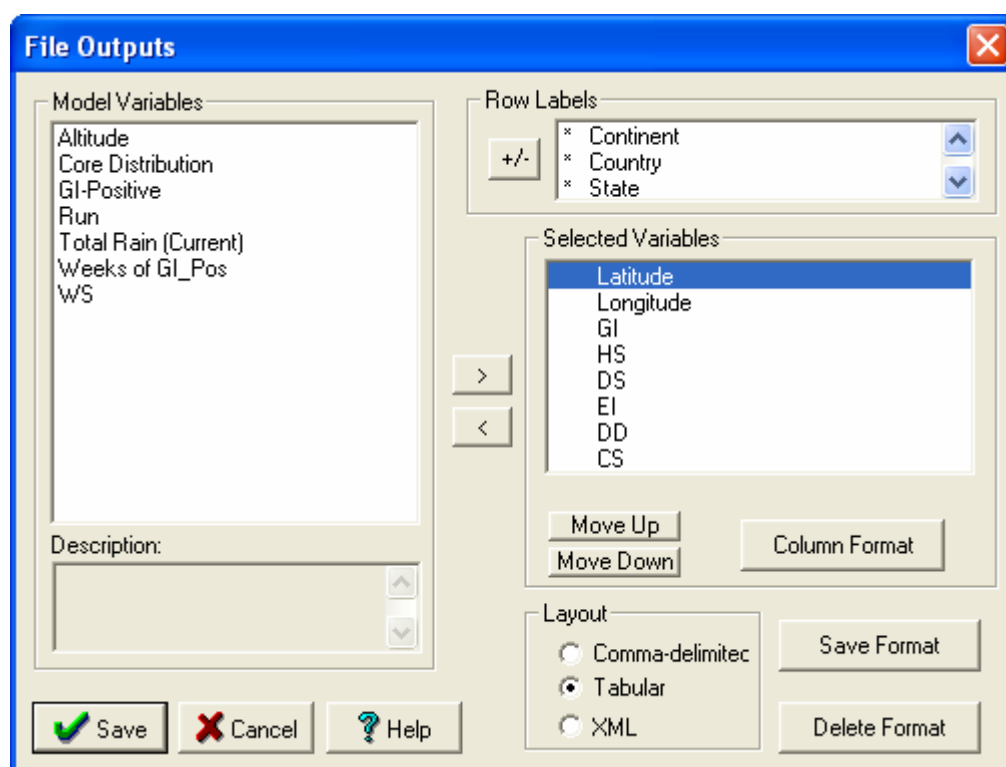


When a map is displayed on the screen, it is possible to print it using the File menu's **Print** or **Print Preview** options. Before printing the map, go to **Print Setup** (also on the **File** menu) and change the settings to your preferences. One of the choices in the Print Setup dialog is to print the map to fit the page or to print it at a particular scale. If the later option is chosen, a map may spread across more than one page. Selecting **landscape** orientation will often allow the map to better fill the page. In the printed map, various dashed and dotted lines replace the coloured lines. The Zoom button in the Print Preview can be used to examine the appearance of the map as it will be printed in some detail.

## 28.4 Creating Text Files

It is possible to save output from simulations as a text file. The method for formatting the text file is nearly the same as that for creating a table and file formats can be saved and reused for later files. The layout for the text file must be set and there are choices of having each column of data separated by commas or arranged in a tabular format. An additional choice allows output files to be written in XML format.

**Fig. 28-28 The text file output dialog box**



➤ **To create a text output file**

- 1.** While the Run Window is open choose **File** from the **Results** menu. If any file formats have been saved previously, a small popup menu appears from which either **New** (to create a file with a new format) or one of the saved formats must be selected. This opens the **File Outputs** dialog.
- 2.** Choose each variable to be added to the file in the order they are to appear and left-click the **Include in File** button.
- 3.** In the **Layout** box, choose either **Use comma delimiters** or **Use Tabular Format**.
- 4.** Type in the name of the output file (or use **Browse** to select a path and filename). Save the format if required (using **Save File Format**) and click on **OK**.

Fig. 28-29 shows the first few lines from a Results file in both the **Tabular** and **Comma-delimited** formats.

**Fig. 28-29** Two example of file output from DYMEX; the upper file is columnar, while the lower is comma-delimited.

Run 1 (09:07) Tough Nut Orchard model; ROC.DAT (1 Jan 71)

(1) Simulation Date  
(2) Minimum Temperature  
(3) Maximum Temperature  
(4) Rainfall  
(5) Egg: Development Time

(1)	(2)	(3)	(4)	(5)
1/1/1971	23.00	38.00	0.00	5.00
2/1/1971	23.00	41.00	0.00	5.00
3/1/1971	26.00	41.00	0.00	5.00
4/1/1971	25.00	37.00	0.00	6.00
5/1/1971	24.00	34.00	0.00	6.00
6/1/1971	24.00	31.00	0.00	6.00
7/1/1971	23.00	32.00	0.00	6.00
8/1/1971	21.00	34.00	0.00	6.00

Run 1 (09:07) Tough Nut Orchard model; ROC.DAT (1 Jan 71)

"Simulation Date","Minimum Temperature","Maximum Temperature","Rainfall","Egg: Development Time"

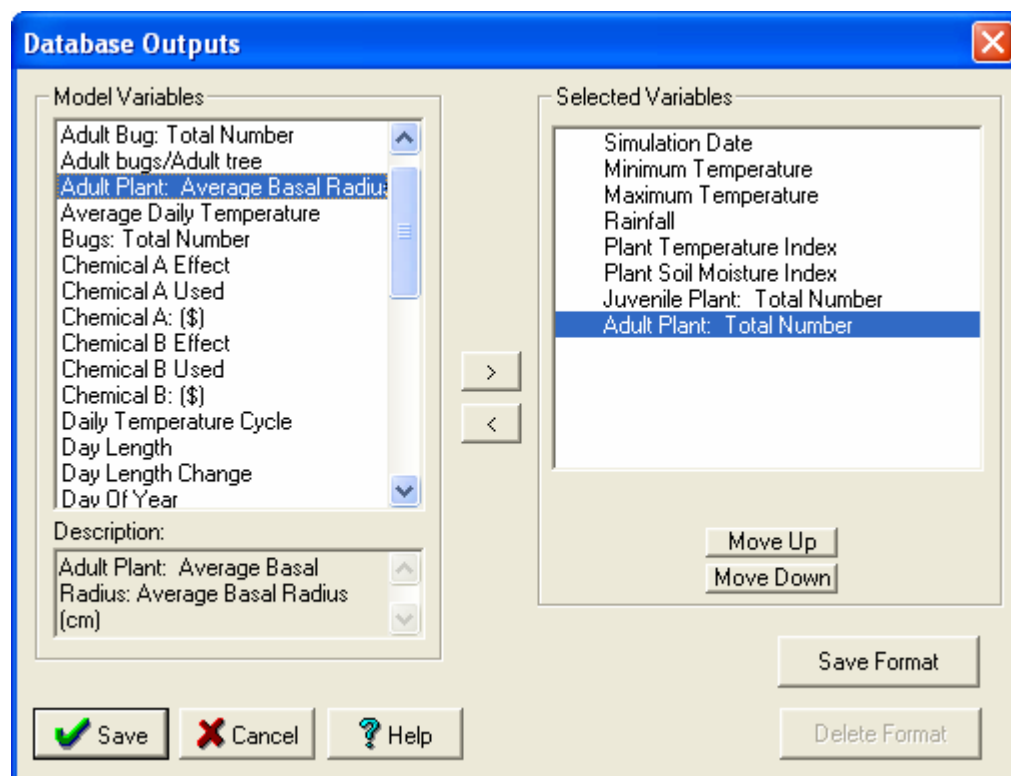
1/1/1971,23.00,38.00,0.00,5.00  
 2/1/1971,23.00,41.00,0.00,5.00  
 3/1/1971,26.00,41.00,0.00,5.00  
 4/1/1971,25.00,37.00,0.00,6.00  
 5/1/1971,24.00,34.00,0.00,6.00  
 6/1/1971,24.00,31.00,0.00,6.00  
 7/1/1971,23.00,32.00,0.00,6.00  
 8/1/1971,21.00,34.00,0.00,6.00

## 28.5 Creating Database Tables

DYMEX results can be easily copied directly to a database table which can be opened using Microsoft Access®. Once a simulation is complete select **Database** from the **Results** menu. If one or more database table formats have been saved from earlier runs, a popup menu appears, allowing a choice of format. Choose **New** to create a new table format, or select the format required. The **Database Outputs Specification** dialog (Fig. 28-10) can then be used to specify which variables to include in the table. If the output was

produced by a **Multiple** run, the row labels are automatically included into the table.

**Fig. 28-30** The database output dialog box.



## 29 The Map Manager facility

The DYMEX **Map Manager** is used to create *Region Definitions* that are then used as the base onto which the data layers of the map are drawn. Map features specified by the Map Region definition include the boundaries of the region, the included shape files that determine the geographical features visible on the map and the map projection. Before any DYMEX output variable can be mapped, the appropriate Region Definition must have been created. Region Definitions are stored in *Region Definition Files* (.mdf) in the Map Directory, as specified in the **Operating Preferences** (Section 25.1).

To understand how a display map in DYMEX is related to the *Region Definition*, it is necessary to understand how such a map is structured. A map consists of a number of layers, as follows:

Data Layer  
Shape File n  
...  
Shape File 1  
Background

This is illustrated in Some shape file data obtained from **CIP** (Centro Internacional de la Papa) with their permission have been included for use with

DYMEX. Visit the CIP website (<http://www.diva-gis.org/Data.htm>) to download the most current version of the data. Other shape file data (such as separate shape files for each continent) is also available from that site.

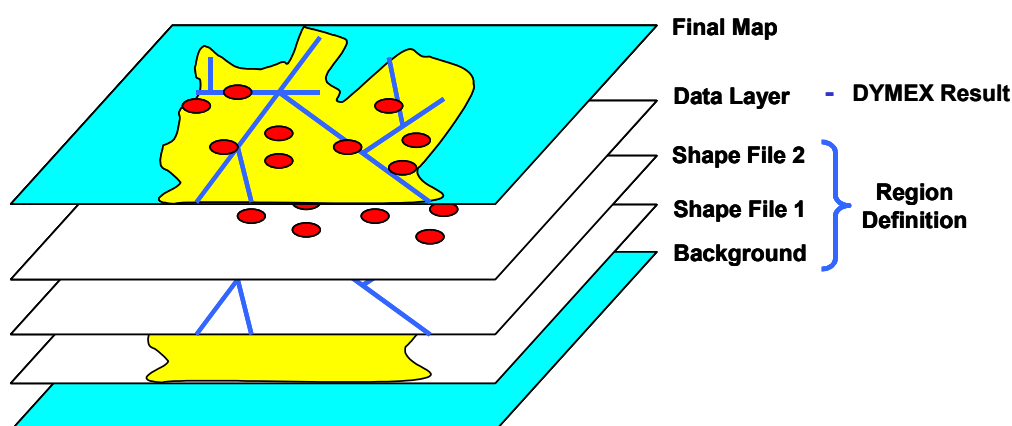
Fig. 29-1. The final map produced when selecting the “**Results|Map**” menu item after a run of DYMEX is the result of drawing all these layers (in order, from bottom to top), within the coordinate boundaries specified and with the specified map projection. Every layer except the Data layer is defined in the *Region Definition*. The Data Layer is added from the specifications provided in the *Map Format* (set in the **Map Specification** dialogs, see Section 28.3).

The Background layer is used to provide a background colour for the map. Each Shape File layer is associated with a single file containing spatial data in the ESRI® Shapefile Format (see the *ESRI Shapefile Technical Description* at [www.esri.com/library/whitepapers/pdfs/shapefile.pdf](http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf)). Another layer, not shown in Some shape file data obtained from CIP (Centro Internacional de la Papa) with their permission have been included for use with DYMEX. Visit the CIP website (<http://www.diva-gis.org/Data.htm>) to download the most current version of the data. Other shape file data (such as separate shape files for each continent) is also available from that site.

Fig. 29-1, contains the positions of the title, legend and scale marker. This information is also stored in the *Region Definition*, though the actual contents of the title and legend are added when the map is actually drawn, again from information provided in the *Map Format*.



Some shape file data obtained from CIP (Centro Internacional de la Papa) with their permission have been included for use with DYMEX. Visit the CIP website (<http://www.diva-gis.org/Data.htm>) to download the most current version of the data. Other shape file data (such as separate shape files for each continent) is also available from that site.



**Fig. 29-1 The layers that constitute a map displaying a variable in DYMEX, and the map that is produced from these layers.**

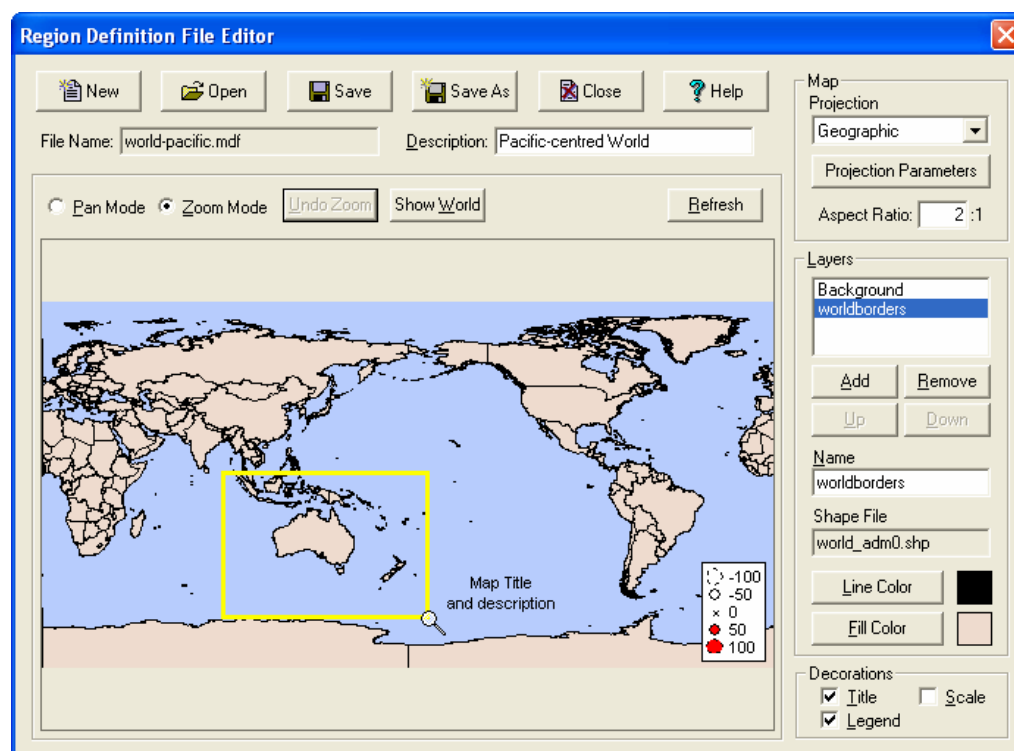
When the MapManager menu item is selected from the File menu, the **Region Definition File Editor** dialog is opened (Fig. 29-2). This is a rather complex dialog, which allows the user to create a Region Definition File by adding the required shapefiles layers to a background, specifying colours for each layer, and selecting a projection for display. The large display panel at the lower left

shows the map as it will be shown (but without the data layer). The row of controls above this panel is used to pan and/or zoom the map until it is cropped to the required coordinates and to select label, title and scale marker decorations.

➤ **To create a new Map Region File**

1. Select the **MapManager** item from the **File** menu.
2. This will open the **Map Definition File Editor** dialog (Fig. 29-2), with only a background layer present in the **Layers** list at the right. Change the background colours, if required, by clicking on the **Line Colour** or **Fill Colour** buttons, and then choosing the desired colours.

**Fig. 29-2** The Region Definition File Editor dialog, showing a map of the world, using the *Geographic* projection with *Central Meridian* set to 180°.



3. Add a Shapefile layer by clicking on the **Add** button, and navigating to the required shape file. Note that shape files generally have the extension **.shp**.



Note that if shapefiles should be stored in the **Map** directory, which is specified in the **Operating Preferences** dialog (Section 25.1).

4. To see the effect that the addition of the layer has on the appearance of the map, click on the **Refresh** button, which redraws the map with current settings applied. You can change the colours of the currently selected layer using the **Line Colour** and **Fill Colour** buttons.
5. Add more layers as required by repeating steps 3 and 4. If a layer has been added by mistake, it can be removed by selecting it and clicking on

the **Remove** button. Similarly, the ordering of layers can be changed with the **Up** or **Down** buttons (however, the background layer will always be the first).

6. If a projection other than the default **Geographic** projection is required, click on the small arrow button to the right of the projection name and select the required projection from the drop-down list. Then set the projection parameters as required using the **Set Parameters** button.



The choice of a projection and different projection parameters can make a big difference to the appearance of a map. If you are unfamiliar with projections, you may wish to consult a reference such as Maling, D.H., *Coordinate Systems and Map Projections*, 2<sup>nd</sup> ed, rev., Woburn, MA: Butterworth-Heinemann, 1993 or Mulcahy, K., "The Map Projection Home Page." at <http://everest.hunter.cuny.edu/mp/>.

7. Click on the Refresh button to update the map display, then use the **Pan** and **Zoom** options to size and position the map to your exact requirements.
8. By default, the map displays a label, legend and scale marker. If any of these are not required, uncheck the corresponding box in the **Decorations** panel at the lower right of the dialog. These 'decorations' can be moved to any desired spot on the map by clicking on them and 'dragging' with the mouse button down.
9. When the map region display shows all features exactly as they are required in the final map, the map region can be given a description by typing the appropriate text into the **Description** box.
10. Click on Save to save the *Region Definition* as a file with the **mdf** extension.
11. Close the dialog using the **Close** button.

## 29.1 Map Region Manipulations

The map produced by the *Region Definition* is shown in the large panel that takes up most of the dialog. Most of the controls in the dialog are used to alter the Region Definition. To avoid lengthy delays after settings are changed on slower computers, the display is not updated every time a change is made to the Region Definition. The **Refresh** button is provided for this purpose.

The shape of the cursor indicates the currently set operation, either "pan" (☞) or "zoom" (🔍). When in pan mode, clicking on the map and moving the mouse with button down will move the map into the same direction as the mouse. In zoom mode, two methods can be used to zoom in onto a point or region. Left-clicking on a map point causes the map to be redrawn at an about 50% larger scale, thus showing a smaller area, positioned so that the point that was clicked stays in the same absolute position on the screen. Similarly, right-clicking on a point draws a 50% smaller version of the map, again positioned so that the clicked point does not move. Note that there are limits to the enlargement and reduction allowed for a map. An alternative way to zoom into an area is to left-click on the point that is required as the top-left corner of

the zoomed area and, leaving the mouse button down, move the mouse to the desired lower-right corner of the area. A yellow rectangle outlining the zoomed area will appear as the mouse is moved on the screen (Fig. 29-2). When the mouse button is released, the area outlined by the zoom-rectangle will be drawn to occupy the full size of the map display area. Zoom operations can be reversed by clicking the **Undo Zoom** button, which takes the map back to the state it was in before the previous zoom operation. **Undo Zoom** can be invoked as often as required to reverse multiple zoom operations.

The **Show World** button changes the boundaries of the region to show the whole world (i.e.,  $-90^{\circ}$  to  $+90^{\circ}$  latitude,  $-180^{\circ}$  to  $+180^{\circ}$  longitude).

Map Region Definitions can contain up to three extra features (decorations). A legend, a title and a scale marker can be specified by checking the appropriate box in the **Decorations** panel. The actual contents of both the legend and title are determined by the data plotted on the map (a “dummy” content is shown in the map panel). Each of these decorations can be “dragged” to the required position using the mouse.

The buttons along the top of the dialog allow a new *Map Region File* to be created, an existing one to be opened, or the one currently being worked on to be saved (**Save**) or saved with a different name (**Save As**). The Close button closes the Region Definition File Editor, prompting the user to save the file if it has been altered.

### 30 The Meteorological Data Manager facility (MetManager)

The *MetManager* program is a Microsoft® Access application that can be used to store and manipulate long-term average meteorological data that is then accessible to DYMEX through the **MetManager** module (see Section 7). For the rest of this section, any reference to *MetManager* will be to the *MetManager* program.

The MetManager stores its data in Access databases with the file extension “.mm”. The location’s data is identified by a location name and the location’s longitude and latitude. The following fields are stored for each location:

<b><i>Continent</i></b>	The name of the continent that the location is in.
<b><i>Country</i></b>	The name of the country that the location is in.
<b><i>State</i></b>	The name of the state that the location is in.
<b><i>Location Name</i></b>	The name of the location
<b><i>Longitude</i></b>	Longitude of the location (in degrees)
<b><i>Latitude</i></b>	Latitude of the location (in degrees)
<b><i>Elevation</i></b>	Elevation (altitude) of the location (in metres)
<b><i>Importance Level</i></b>	An integer (0-5) that is used to separate sets of closely-spaced locations
<b><i>Minimum Temperature</i></b>	The <u>Average Daily Minimum Temperature</u> (12 monthly values or 52 weekly values)
<b><i>Maximum Temperature</i></b>	The <u>Average Daily Maximum Temperature</u> (12 monthly values or 52 weekly values)
<b><i>Rainfall</i></b>	The <u>Total Rainfall</u> (12 monthly values or 52 weekly values)
<b><i>9am RH</i></b>	The <u>Average Daily 9am Relative Humidity</u> (12 monthly values or 52 weekly values)
<b><i>3pm RH</i></b>	The <u>Average Daily 3pm Relative Humidity</u> (12 monthly values or 52 weekly values)

Several user-defined fields are also available for storing such variables as, for example, Evaporation, but these are not accessible for use by the current version of DYMEX.

The database can contain data for a large number of locations, and a mechanism for retrieving or accessing a subset of these locations has been provided. A ***Location Selection*** is the set of locations each of which satisfies a specified number of conditions (the ***Selection Criteria***). For example, if the Selection Criteria are that (1) the Country is *Australia* and (2) the Latitude is



less than  $-30^{\circ}$ , then the Location Selection will include all the Australian locations in the database whose latitude is less than  $30^{\circ}$ . Many Selection Criteria can be defined, named and stored within the database. DYMEX uses the resulting Location Selections when running MetManager sequences (see Section 27.3). Note that the term **Location Selection** (or **Selection**) is commonly used for both a set of locations as well as the **Selection Criteria** that define that set.

The MetManager is started by clicking on the MetManager icon in the DYMEX Program Group. This displays the MetManager main menu, as shown in nager operations.

Fig. 30-1. Note that the “menu” will be shown within a Microsoft Access window. This window has its own menu bar, which however, is not relevant to MetManager operations.

**Fig. 30-1 The MetManager Main Menu.**



The “menu” consists of a set of buttons and their descriptions. The name of the current data-file is shown at the bottom of the menu. If no data file has been specified, this must be done before any other actions are possible (either by creating a new data-file, or by opening an existing one).

### 30.1 Create/Edit/Delete Location Selections

The **Create/Edit/Delete Location Selections** option is used to create, edit or delete stored Location Selections (*Selection Criteria*). It opens the **Edit**

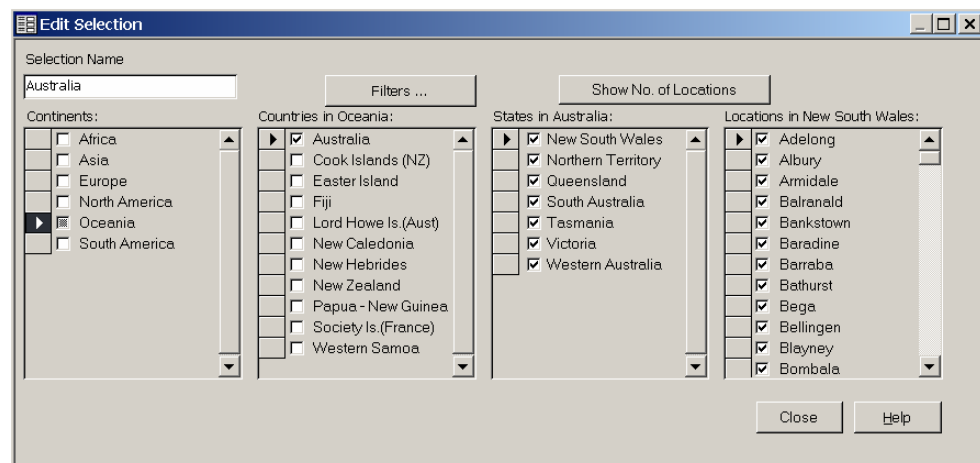
**Selections** dialog, which list all of the currently available selections (Fig. 30-2). To add a new Selection, click on the **Add** button. To edit or delete a Selection, click on the appropriate Selection name in the list to highlight it and then click the **Edit** or **Delete** button. Note that deleting a selection only deletes the *Selection Criteria* (it does not delete any data from the database). To see the number of locations in the database that match the selection criteria, click on **Show No. of Locations**.

**Fig. 30-2 The Edit Selections dialog.**



When a new selection is added, the user is first prompted for the name of the new selection. Use a name that clearly describes the set. The Selections dialog is then presented (the **Edit** button goes directly there), where the *Selection Criteria* are specified (Fig. 30-3).

**Fig. 30-3 The Selections dialog, used for specifying the Selection Criteria.**



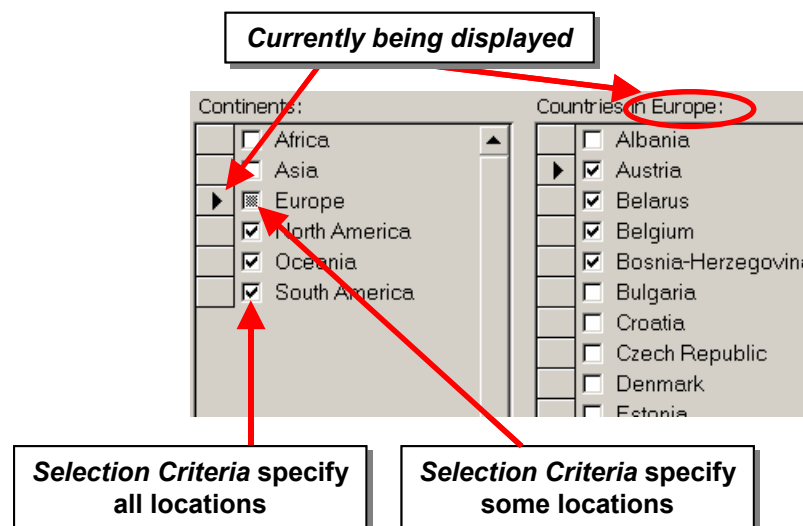
The **Selection** dialog is rather complex. It contains four lists that show *Continents*, *Countries*, *States* and *Locations*. To understand its operation

correctly it needs to be understood that the dialog is actually showing two separate things within its four list panes, as follows:

1. A view into the contents of the database as a hierarchical structure organized by Continent, Country, State and Location
2. The *Selection Criteria* that use Continent, Country, State or Location names.

Considering the first of these, the **Continent** pane lists all of the continents represented within the database. Clicking on the grey, rectangular panel in front of a continent's name will "select" that continent (indicated by an arrow symbol, ►), and show the countries belonging to that continent in the **Country** pane. Again, an arrow symbol is used to show the currently selected country, whose states are shown in the **States** panel, while the same symbol in the States panel shows the currently selected state, whose constituent locations are shown in the **Locations** pane. If a country has no states defined, selecting that country will display its locations (with an empty **State** pane).

Fig. 30-4 A part of the Selections dialog, showing Europe selected for display.

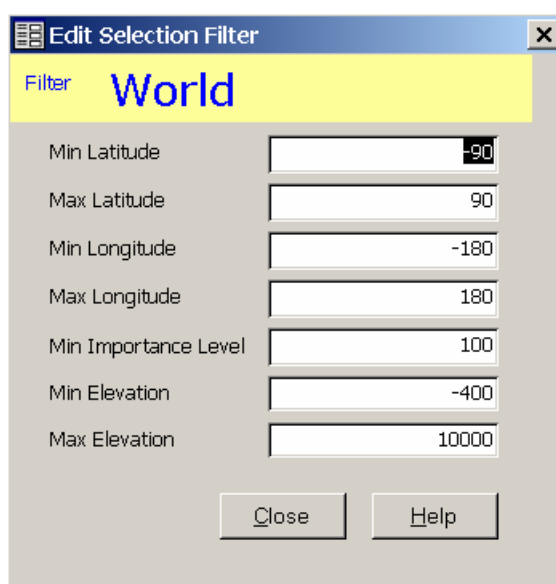


Whether a particular entity (continent, country, state or location) is included in the *Selection Criteria* being defined is indicated in the checkbox next to the entity's name. A tick indicates that the entity is completely included. For example, the tick next to *South America* in Fig. 30-4 indicates that the *Selection Criteria* include ALL South American locations. Note that this does not mean that the set of locations specified by the criteria (the **Location Selection**) will include all South American locations – there may be another component of the *Selection Criteria* that (for example) specifies that locations are to have an altitude less than 2000m. The grey square within the checkbox next to *Europe* indicates that Europe is partially included in the Selection Criteria. This can be seen by

examination of the **Countries** pane, where only a few of the countries are selected.

Entities are added to the *Selection Criteria* by clicking on the checkbox so that it shows a tick. Entities are deselected by clicking until the checkbox is empty. An entity that is only partially selected can only be unselected by removing the included, lower-level entities from the *Selection Criteria*. Other items (such as latitude limits) included in the Selection Criteria can be changed by clicking on the **Filters** button (Fig. 30-3) to obtain the **Edit Selection Filter** dialog (Fig. 30-5).

**Fig. 30-5 The Selection Filter dialog.**



The **Edit Selection Filter** dialog specifies other criteria that a location must conform to in order to be included in the set of locations making up the *Location Selection*. The latitude and longitude limits can be used to set a bounding rectangle within which locations must lie, while the elevation limits set an elevation band. The **Min Importance Level** can be used to limit locations to those with a specified level of “importance”.

## 30.2 Create a new data-file

This option is used to create a new (empty) **MetManager** database file. It leads to a file dialog, from whence the user can navigate to the appropriate directory and enter the name of the file. The file should be created as a MetManager file (with extension “.mm”). When the new file has been created, it is automatically opened as the current database file within **MetManager**.

## 30.3 Open an existing data-file

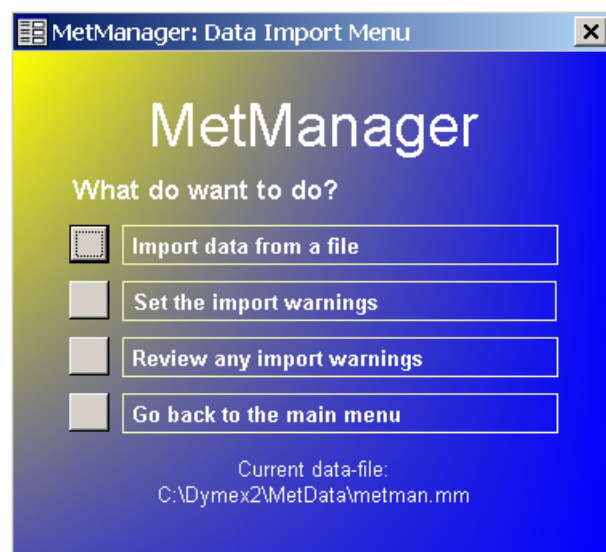
This option is used to open an existing **MetManager** database file. It leads to a file dialog, from whence the user can navigate to the appropriate directory

and enter or browse for the name of the file. MetManager file should have the extension “.mm”.

## 30.4 Import meteorological data

This option is used to add meteorological data to the currently open database from another file. It leads to the Data Import Menu, which provides several choices .

**Fig. 30-6 The MetManager Data Import menu.**



**Import data from a file** – this selection leads to a File Open dialog, where the file from which the data is to be imported can be selected. Three file types are supported for data input. These are (1) BMT files, as used as the database format in Version 1 of the MetManager program, (2) LOC/MET files (also used by Version 1 of the MetManager program as an import format) and (3) another MetManager Version 2 database file (.mm file). The LOC/MET file format is a plain text format, detailed in Appendix 3.

To import new locations, click on the Import Data button of MetManager. The MetManager will ask you if you want to replace any existing data with the corresponding imported data. If you select ‘Yes’ and you have locations in your new data that are already in the database (ie they have the same names for continent, country, state and location), the new data will replace the original data. If you select ‘No’, the original data will be protected and any such duplicate locations will not be imported.

You will then be asked to select the location file (.LOC). Choose the directory where the file is located and select the file by clicking on its name and then the ‘OK’ button. There must also be a file containing the corresponding meteorological data (.MET file) in the same directory. The MetManager will now import the new data into the database.

The structure of your data files is very important (See Appendix 3). Example files of each (bolivia.loc and bolivia.met) are included in the “.\Dymex2\MetManager” directory. Use these files as a template for creating your files. The column setup will need to be EXACTLY the same as these example files otherwise MetManager will not import the data. Generally, an error message during the importation will identify any problems in the files. Note that if problems are found at any point during the import operation, no changes will be made to the database.

**Set the import warnings** – when data is imported into the database, it is checked for validity by comparing its values with a preset range. The allowed range can be set separately for each data field in the **Import Validation** dialog (Fig. 30-7).

**Fig. 30-7 The MetManager Import Validation dialog.**

Field Name	Lower Limit	Upper Limit
Latitude	-90	90
Longitude	-180	180
Elevation	-400	10000
ImportanceLevel	0	5
User1		
User2		
User3		
User4		
User5		
ImportIndex		

A drop-down list at the top of the dialog (labelled “**Validation Rules**”) allows selection of either “Locations” or “Meteorological” data. Selecting either option shows a list of the corresponding data fields (the “Field Name” column), together with each field’s current allowable limits (“Lower Limit” and “Upper Limit”). If a particular limit is blank, no error checking will be performed for the corresponding data in that direction (i.e., “Upper” or “Lower”). The scroll bar at the right can be used to change the displayed fields if there are more fields than can be displayed simultaneously in the dialog.

**Review any import warnings** – after data has been imported into the database, this option allows review of the warnings that were generated during the import process.

## 30.5 Edit location and meteorological data

This option is used to edit the location and meteorological data in the currently open database. It leads to the Edit Data Menu, which provides several choices.

Fig. 30-8 The MetManager Edit Data menu.




**Edit the names of continents** – allows the names of the continents present in the database to be changed. Each continent’s name is shown in a separate edit field. To change a name, select the appropriate field and amend the name. Click on the “Close” button to exit the dialog. The scroll bar at the right can be used to move the list of names up or down if there are more names than can fit into the dialog at once.

**Edit the names of countries** – allows the names of the countries present in the database to be changed. Each country’s name is shown in a separate edit field. To change a name, select the appropriate field and amend the name. Click on the “Close” button to exit the dialog. The scroll bar at the right can be used to move the list of names up or down.

**Edit the names of states** – allows the names of the states present in the database to be changed. Each state’s name is shown in a separate edit field. To change a name, select the appropriate field and amend the name. Click on the “Close” button to exit the dialog. The scroll bar at the right can be used to move the list of names up or down.

**Edit the names of locations** – allows the names of the locations present in the database to be changed. Each locations’s name is shown in a separate edit field. To change a name, select the appropriate field and amend the name. Click on the “Close” button to exit the dialog. The scroll bar at the right can be used to move the list of names up or down. The “Find” button at the top of the dialog can be used to enter the first few characters in the locations’s name,

which helps to locate it more quickly. A “Location Details” button () gives access to some other location properties, which can also be changed there.

**Edit meteorological data** – any location’s meteorological data can be changed by selecting this menu item. The resulting “Edit Meteorological Data” dialog contains a set of selection boxes at the top, from which the desired locations Continent, Country, State and location name can be specified. When this has been done, the locations’s meteorological data is displayed in the dialog. A particular data item is changed by clicking on it and typing in its new value. The value will be recorded to the database when the cursor leaves the data field. Pressing the “Esc” button on the keyboard before the cursor leaves the data field will restore its previous value. The “Edit” button next to the “Location” name can be used to edit other location properties such as its latitude and longitude.

## **30.6 Delete locations**

This option is used to delete locations (and their associated meteorological data) from the currently open database. The locations to be deleted must be specified using a ***Location Selection***, as described in section 30.1. All of the locations specified by the ***Location Selection*** are deleted from the database, as is the ***Location Selection*** itself.

## **30.7 Maintain the database**

**Try to reduce the size of the database** – After many deletion and editing operations, the database may be somewhat fragmented. This option will attempt to compact it so that it takes up the minimum disk space.

**Set up user-defined data** – Several user-defined data fields are available for provision of additional data that describes a location (such as, for example, soil characteristics). Note, however, that this data is not accessible in the current version of DYMEX.



## Appendix 1. The Model Trace File

Using the Model Trace option (see Model Trace Options, page 84), more detailed output can be obtained, especially about the Lifecycle module, than is possible using the Result tables alone. An extract from such a file (the Model Trace File, *dymex.dbg*) is shown on the next page. The extract shows trace information from Step 6 of a simulation run of the example “Beetle” model, using the default parameter values. The maximum trace options were turned on to obtain this output (i.e, all boxes in the **Model Trace Options** dialog, Fig. 25-2, were checked except for the **Show only Lifecycle module** box).

As the model simulation progresses, each module writes information to the Model Trace File. Some modules write only their name (for example, the *Meteorological Data* module). Other modules write the values of the output variables (for example, *Timer* and *Daylength*).

The **Circadian** Module (*Daily Air Temperature*) writes the values of the individual cycle points, as calculated from the cycle shape being used (see *The Circadian module*, page 53).

The **Event** module (*Adult Insecticide Treatment*) writes the current value of the internal variable *Days since Event*, as well as the value of the output variable (*Event Var*). Note that since no event has yet occurred by Step 6, the variable has a large value.

The **Lifecycle** module writes by far the most information to the Model Trace file. Most of this deals with to the dynamics of the cohorts within the lifecycle. Each cohort is numbered, so its fate can be followed. The cohort numbers are often in square parentheses (eg, [26]), and will sometimes be followed by one or more lower-case letters that indicate a cohort property, as follows:

d	a “Dummy” Cohort (see <i>Dummy Cohorts</i> , page 33)
i	a cohort created from a lifestage introduction
u	the cohort is being updated
g	the cohort is producing graduates this timestep
e	the cohort is “empty”, i.e., it meets the Cohort Removal Conditions
w	the cohort is required for calculating a “duration” variable (such as Development Time”, and cannot be deleted, even if it meets the Cohort Removal conditions

The first part of the **Lifecycle** section (starting with “**LIFECYCLE Trace:**”) is a detailed report of the dynamics of cohort creation, updating and deletion. This part of the output will only be present if the **Execution Trace** option in the **Model Trace Options** dialog is selected. The trace is divided into two

blocks, labelled “**#Promoting Graduates**” and “**#Updating Lifestages**”, respectively.

The “**#Promoting Graduates**” block has 3 lines for each graduate cohort that is created at the start of the timestep. The first line shows the source and destination stages (eg, “*Egg->1<sup>st</sup> Larval Stage*”), followed by the values of the Cohort Properties of the graduate “Cohort”. The Cohort Properties are in parentheses, with values listed in the order (Number, Physiological Age, ...), where the “...” indicates user-defined Cohort Properties, if any are present. The second line is written at the creation of a new cohort of the destination stage, and shows the number of individuals in the new cohort (in parentheses). The third line, starting with a “+” symbol, gives the identifier of the new cohort (eg, [25]), the lifestage it belongs to and, in parentheses, its initial number of individuals, and the number of individuals remaining after establishment processes have been applied.

The “**#Updating Lifestages**” block shows any new cohorts that were created without transfer from a previous stage (for example by lifestage initialization or as “dummy” cohorts). These new cohorts are shown on lines beginning with a “+” symbol, in the same way as in the preceding section. The rest of the information in the block is grouped under sections labelled “*-Lifestage: Updating Populations*”, where “Lifestage” is the name of a lifestage in the Lifecycle. The line starting with “*Updating Cohorts*” list each cohort in the lifestage, followed by “u” to indicate it is being updated, and “g” if it is generating graduates during the timestep. The line starting with the phrase “*Deleting Cohorts*” list any cohorts that are being deleted during the timestep.

The last part of the **Lifecycle** section provides details of each cohort remaining in the population at the end of the timestep. The 2<sup>nd</sup> line in this part is a header line that labels each of the items on succeeding lines. “[Id]” is the Cohort Identifier, “CA” the Chronological Age”, while “Number”, “Density” and “PhysioAge” are the corresponding Cohort Properties. If other Cohort Properties are defined (such as “*Stress*” in the example), these will be listed also. The lines starting with a Cohort Identifier in square parentheses then list the corresponding Cohort Properties under that label. A cohort labelled [Grad], if present, contains the current graduates form the lifestage.

```
***** STEP 6 *****
MODULE: Timer                      Day = 36; Day of Year = 36; Simulation Date = 5/2/1965
MODULE: Latitude
MODULE: Daylength                  Daylength = 13.272
MODULE: Meteorological Data
MODULE: Daily Air Temperature
25.786 22.679 20.404 19.571 20.404 22.679 25.786 28.893 31.167 32.000 31.167 28.893
MODULE: Evaporation                Evaporation = 50.754
MODULE: Soil Moisture              Soil Moisture = 0.355
MODULE: Adult Insecticide Treatment Days since Event = 100000; Event Var = 0.0000
```

LIFECYCLE Trace:

```
#Promoting Graduates
Egg->1st Larval Stage (46.8750, 2.23714, 0.24060)
..Creating graduate cohort (46.87500)
+[25], 1st Larval Stage (46.87500, 46.87500)

#Updating Lifestages
+[26](d), Egg(1.00000, 1.00000)
-Egg: Updating Populations
Updating Cohorts - 26u,21u,16ug,0ug,.
Deleting Cohorts - .
-1st Larval Stage: Updating Populations
Updating Cohorts - 25u,20u,15ug,.
Deleting Cohorts - .
+[27](d), 2nd Larval Stage (1.00000, 1.00000)
-2nd Larval Stage: Updating Populations
Updating Cohorts - 27u,22u,17u,12u,7u,2u,.
Deleting Cohorts - .
+[28](d), 3rd Larval Stage (1.00000, 1.00000)
-3rd Larval Stage: Updating Populations
Updating Cohorts - 28u,23u,18u,13u,8u,3u,.
Deleting Cohorts - .
+[29](d), Pupa(1.00000, 1.00000)
-Pupa: Updating Populations
Updating Cohorts - 29u,24u,19ug,.
Deleting Cohorts - 19,.
-Pre-emergence Adult: Updating Populations
Updating Cohorts - .
Deleting Cohorts - .
-Free-living Adult: Updating Populations
Updating Cohorts - .
Deleting Cohorts - .
```

MODULE: Beetle Lifecycle - Cohort Details:

```
[Id] CA: Number ( Density; PhysioAge, Stress )
*** Egg
[26] d 1: 1.00 ( - ; 0.4736, 0.0434 )
[21] d 2: 1.00 ( - ; 0.8979, 0.0828 )
[16] de 3: 0.00 ( - ; 1.3350, 0.1150 )
[10] de 3: 0.00 ( - ; 1.3243, 0.1495 )
[ 5] de 3: 0.00 ( - ; 1.3629, 0.1621 )
[ 0] 6: 3.91 ( - ; 2.7107, 0.2840 )
[Grad] 11.72
*** 1st Larval Stage
[25] 1: 46.88 ( - ; 0.4114, 0.2406 )
[20] 2: 187.50 ( - ; 0.7859, 0.2012 )
[15] 3: 187.50 ( - ; 1.1700, 0.1690 )
[Grad] 562.50
*** 2nd Larval Stage
[27] d 1: 1.00 ( - ; 0.0947, 0.0000 )
[22] d 2: 1.00 ( - ; 0.1796, 0.0000 )
[17] d 3: 1.00 ( - ; 0.2670, 0.0000 )
[12] d 4: 1.00 ( - ; 0.3596, 0.0000 )
[ 7] d 5: 1.00 ( - ; 0.4521, 0.0000 )
[ 2] d 6: 1.00 ( - ; 0.5421, 0.0000 )
*** 3rd Larval Stage
[28] d 1: 1.00 ( - ; 0.0947, 0.0000 )
[23] d 2: 1.00 ( - ; 0.1796, 0.0000 )
[18] d 3: 1.00 ( - ; 0.2670, 0.0000 )
[13] d 4: 1.00 ( - ; 0.3596, 0.0000 )
[ 8] d 5: 1.00 ( - ; 0.4521, 0.0000 )
[ 3] d 6: 1.00 ( - ; 0.5421, 0.0000 )
*** Pupa
[29] d 1: 1.00 ( - ; 0.4736, 0.0000 )
[24] d 2: 1.00 ( - ; 0.8979, 0.0000 )
```

## Appendix 2. The LOC/MET File Format

The two files hold the location data (.LOC files) and the meteorological data (.MET files) respectively. Each of these files can be in either a fixed-field or comma-delimited format. The MetManager automatically detects whether a file is in fixed-field or comma-delimited format.

### (a) Fixed field format

An example of the \*.loc file (ie bolivia.loc) is given below and could be used to add two new locations (Chulumani and Patacamaya) in Bolivia.

```
*1.00 SOUTH AMERICA
*1.01 Bolivia
#*      nostates
0      Chulumani          16.4 S   67.5 W 1580
0      Patacamaya         17.2 S   67.9 W 3789
```

The file starts with a header that includes the continent, country, and optionally, a state. Note that these items appear on separate lines of which the first two are prefixed by a \* and four numeric characters followed by a space and the name of the continent or country. \*1.00 denotes continent and 1.xx (where xx is any non zero integer) denotes country. The optional line for state is prefixed by # \* and four spaces followed by the name of the state or 'nostates', where state definitions do not exist (the state line may be omitted altogether if you do not intend to use states for the entry of new locations). The name of the continent MUST be one of the continents in the CLIMEX database, or the import will fail. The name of the country and state entered should correspond to one already being used in the CLIMEX database. If an unknown name is used for a country or state, a new country or state with that name will be created and the new locations will not be added to that respective continent and/or country and/or state. Be particularly careful with spelling of these items to avoid errors.

Following the header on a new line appears the **Location Data** for each new location. This includes information about the importance level, the name of the location, its coordinates and elevation (in metres. The Location Data is of a fixed format with data in columns with each location line specified as below (for columns not referred to, insert a normal space).

```
1      Importance level code (0-5; blank=0)
7-25   Location name
26-29  Latitude
31     North/South indicator (N or S)
34-38  Longitude
40     East/West indicator (E or W)
41-45  Elevation (in metres)
```

The **Meteorological Data** file (example below) holds all the climate data for each location. For each month (1 - 12) the file holds average maximum daily temperature, average minimum daily temperature, average monthly rainfall,

average daily relative humidity at 9 am, average daily relative humidity at 3 pm and the 10-character location code defined in the Location Data file above. Note in the example below that the 3pm Relative Humidity values are all zero. CLIMEX interprets these as being absent, and calculates them using the 9am value (9am RH x 0.85). The Meteorological Data file is given a .met file extension (ie bolivia.met).

1	26.9	15.9	171	12.9	.0	CHULUMANI
2	26.8	15.8	223	12.5	.0	CHULUMANI
3	26.9	15.5	141	12.1	.0	CHULUMANI
4	26.9	14.7	56	11.6	.0	CHULUMANI
5	25.4	14.6	38	11.2	.0	CHULUMANI
6	24.5	11.5	17	11.0	.0	CHULUMANI
7	24.7	11.3	20	11.1	.0	CHULUMANI
8	26.3	12.3	58	11.5	.0	CHULUMANI
9	27.0	13.4	97	11.9	.0	CHULUMANI
10	27.7	14.9	99	12.3	.0	CHULUMANI
11	28.5	15.1	117	12.7	.0	CHULUMANI
12	27.5	15.7	188	13.0	.0	CHULUMANI
1	22.7	5.8	88	12.9	.0	PATACAMAYA
2	21.9	6.0	69	12.5	.0	PATACAMAYA
3	21.1	4.7	47	12.1	.0	PATACAMAYA
4	19.3	2.1	15	11.6	.0	PATACAMAYA
5	16.5	-1.6	8	11.2	.0	PATACAMAYA
6	14.6	-4.8	2	11.0	.0	PATACAMAYA
7	14.9	-4.7	1	11.1	.0	PATACAMAYA
8	16.8	-2.6	9	11.4	.0	PATACAMAYA
9	19.1	1.5	37	11.9	.0	PATACAMAYA
10	22.6	3.0	18	12.4	.0	PATACAMAYA
11	24.3	4.5	33	12.8	.0	PATACAMAYA
12	22.6	5.7	76	13.0	.0	PATACAMAYA

The met file data must contain the same number of locations and be in the same order as the loc file entries. The format of the file is as follows (columns not used are spaces):

1-3	Month
5-9	Average Daily Maximum Temperature
11-15	Average Daily Minimum Temperature
17-21	Average Monthly Rainfall (mm)
23-27	Average Daily 9am Relative Humidity (%)
29-33	Average Daily 3pm Relative Humidity (%)

### **(b) Comma-delimited format**

The Continent, Country and (optional) State lines start with a # symbol in this format. The lines that correspond to a location must contain the following items, separated by commas:

Latitude, Longitude, Elevation, Location Name, Importance level

Note the "Location Name" may not contain a comma. An example of a comma-delimited .loc file is shown below:

```
#Continent, Africa
#Country, Algeria
#State, nostates
32.8,-0.6,1072,Ain-Sefra,0
36.8,3,50,Algiers,0
```

The .MET file contains the following data items, each separated by a comma from the next item:

Month, Maximum Temperature, Minimum Temperature, Rainfall, 9am Relative Humidity, 3pm Relative Humidity

An example of a comma-delimited .MET file containing the data for two locations is shown below:

```
1,12.2,-0.6,10,79,52
2,15.6,1.1,10,76,50
3,17.2,3.9,15,68,36
4,23.3,7.8,10,58,29
5,27.2,11.1,15,59,29
6,32.8,15.6,28,46,26
7,37.8,19.4,8,39,18
8,35.6,18.9,8,46,21
9,32.2,15.6,15,51,29
10,24.4,9.4,28,68,34
11,16.1,5,28,77,46
12,12.8,1.1,18,78,48
1,15,9.4,112,75,66
2,16.1,9.4,109,72,60
3,17.2,11.1,74,71,59
4,20,12.8,41,67,57
5,22.8,15,46,72,60
6,25.6,18.3,15,72,60
7,28.3,21.1,3,73,60
8,29.4,21.7,5,70,60
9,27.2,20.6,41,74,62
10,23.3,17.2,79,72,60
11,18.9,13.3,130,73,63
12,15.6,10.6,137,72,64
```

**Index**

- Access database, 52, 140, 146
- Auxiliary Parameter File, 77
- Basal Evaporation, 71
- Chart, 18, **109–20**, 115, 118
  - Axis, 110, 113–15, 115–17
  - Box Axis, 114, 115
  - Create, 110
  - Export, 119
  - Font Scaling, 113
  - Format, 110, 112
  - Menu, 17, 118–19
  - Panel, 112, 115
  - Printing, 120
  - Series, 111, 117–18
  - Zoom, 119
- Chronological Age, 26, 41, 42, 156
- Cohort, 17, **25–29**, 29, 30, 31, 32, 33, 41, 42, 43, 44, 85, 155, 156
  - Dummy, 33, 44, 155
- Cohort Duration, 5
- Cohort Property. *see* Cohort Variable
- Cohort Removal, 17, **42–44**
- Cohort Variable, 26, **41–42**
  - Scope, 41
  - Update Method, 30, 42
- Combination Rule, 8, 29, 80
- Complement Product, 31
- Component Window, 75, 84, 87, 90, 94
- Cost per instance, 61
- Current Average, 30
- Current Value, 30
- Cycle Shape
  - Composite (Sine+Exponential), 54
  - Composite (Sine+Sine), 54
  - Sine, 54
- Data File
  - Date Format, 48
- Data File Format, 45
- Data File Reader. *see* Module type: DataFile
- Data Timestep, 45, 51
- Data Validity Checks, 50
- DataBase Output, 140
- Day of Year, 23
- Daylength, 54, 72
- Daylength Change, 72
- Days since Event, 61
- Days since Start, 23, 111
- Degree-day, 67
- Delay Variables, 5
- Density, 41, 42, 156
- Development, 8, 24, 26, 29, 31, 33, 53
- Development Time, 4, 5, 33, 155
- Dialog
  - Chart Series Format, 117
  - Chart Specification, 110
  - Chart X-Axis, 113
  - Chart Y-axis, 116
  - Cohort Removal, 43
  - Column Format, 121
  - Event Calendar, 62
  - File Format, 47
  - Grid Map, 134
  - Label Map, 133
  - Map Association, 126
  - Map Type, 126
  - Model Description, 18
  - Module Information, 21
  - Operating Preferences, 83
  - Parameter Sequence, 102
  - Sequence Types, 102
  - Summary Variables, 75
  - Symbol Map, 127
  - Table Specification, 120, 122
  - Threshold Event, 65
  - Timer initialization, 24
- Direct Update, 30
- Dispersal, 10, 25, 29, 31–**32**
- Drainage Rate, 70
- Dymex Style. *see* Map, Zoom Style
- Environment, 83
- Equatorial Region, 73, 89
- Equilibrium, 23, 90, 91
- Equilibrium Variable, 23
- Error Handling, 14
- Evaporation, 45, 146
- Evapotranspiration coefficient, 71
- Event
  - Disable, 63
- Event Cost, 61, 62
- Event Termination, 61
- Event Threshold, 61, 63

- Event Variable, 62
- Excel
  - saving tables to, 124
  - using data from, 46
- Execution, 17, 91
- Factor, 7
- Fecundity, 25, 33
- Function, 8, 80
- Genetic Mixing, 32
- GIS Style. *see* Map, Zoom Style
- Global Scope, 26, 42
- Graduates, 28, 155, 156
- Immigration, 3
- Import data, 151
- Import formats, 158
- Injector, 34, 38
  - Multi-cohort, 36
- Installation, 4
- Latitude, 59, 89, 125
- Lifecycle, 8, 41, 84, *see* Module, Lifecycle
  - Branch, 3, 32, 39
  - Multiple, 42
- Lifecycle Diagram, 24, 38
- Lifecycle Menu, 38
- Lifecycle Window, 38
- Lifestage, 2, 3, 5, 17, 25, 32, 33, 35, 38, 41, 43, 85, 155, 156
  - Process, 29–33
- Lifestage Window, 40
- Local Scope, 42
- Location Selection, 101, 146
- Location Selection \b\r
  - Location\_Selection, 147
- Longitude, 89, 125
- Map, 84, **125–39**, **141–45**
  - Association, 126
  - Colour-graded, 134
  - Create, 126
  - Format, 142
  - Grid, 134
  - Label Type, 133
  - Legend, 145
  - Menu, 17
  - Projection, 138, 144
  - Region Definition, 125, 128, **141**, 144
  - Symbol, 127
  - Zoom, 137, 144
  - Zoom Style, 84
- Map Manager, **141–45**
- Menu
  - Results, 120
- MetManager format, 158–60
- MetManager Program, 101, **146–54**
- Missing Values, 49
- Mnemonic, 124
- Model Component Window, 14, **15–17**, 19
- Model Description, 17
- Model Description (GMD) file, 10, 11, 14, 76, 88
- Model Help File, 88
- Model Timestep, 14, 45
- Model Trace File, 155–57
- Model Trace Option, 17, 28, 85
- Module, 2, 3, 5, 6, 7, 8, 13, **21–23**, 23, 57, 61, 62, 68, 76, 80, 85, 90, 93, 100, 155
  - Initialization, 22
  - Run Options, 44
  - Settings, 21
  - Summary, 75
  - Summary Manager, 75
- Module type
  - Accumulator, 3, *see* Running Mean
  - Adjustable Circadian, 3
  - Circadian, 20, **53–55**, 155
  - Circadian (adjustable), 55
  - Climate Change Scenario, 3, 20, 73, 89
  - CLIMEX, 89
  - Counter, 3, 20, 68
  - DataFile, 3, 6, 20, **45–50**, 58, 91, 99, 125
  - Daylength, 4, 20, 72
  - Degree-day, 67
  - DemeSplitter, 73
  - DemeStatistics, 74
  - Equation, 20, 68, 87
  - Evaporation, 6, 20, 71, 72
  - Event, 3, 20, **60–65**, 96, 97, 155
  - Expression, 5, 7, 20, **66**, 87
  - Function, 20, **67**, 87
  - Lifecycle, 20, **24–44**, 155
  - MetBase, 20, 45, 47, 51, 58, 91, 99
  - MetManager, 3, 20, 52, 101, 125, 146, 147, 152, 158



- QueryFile, 20, 56, **58**, 100
- QueryUser, 20, 56, 57
- QueryUser/Discrete, 3, 56, **57**
- Running Mean, 20, **67**
- Soil Moisture, 7, 20, **71–72**
- Storage, 3, 20, **69–70**
- Summary Manager, 16, **74**
- Switch, 3, 20
- Timer, 20, **23–24**, 34, 90, 91, 111
- Weather, 3
- Mortality, 8, 26, 29, 30, 43
- Multiple Run, 122
- Number, 25, 26, 29, 30, 33, 41, 156
- Parameter, 2, 8, 11, 14, 18, 29, **76–82**, 88
  - Default Set, 77
  - File, 76, 77
  - Tree Window, 77
- Parameter Grid, **81–82**, 88
- Parameter Set, 79
  - Description, 80
- Physiological Age, 5, 25, 26, 29, 30, 31, 41, 44, 156
- Preferences, 17
  - Current Model, 86
  - Operating, 17, 83
- Print Preview, 19
- Printing
  - Chart, 120
  - Map, 139
  - Model, 19
  - Parameters, 19, 81
  - Table, 125
- Process, 8, 29, 40, 80
- Process Component, 8, 29, 40
- Progeny Production, 25, 33
- Proportional Update, 30
- Quick Graph, 124
- Quick Selector Button, 16, 95, 100
- Relative Humidity, 72
- Reproduction, 28, 33
- Residual Fecundity, 25, 26, 33
- Resource Variable, 42
- Run Description, 86
- Run Description Macro, 86
- Run Identifier, 113
- Run Sequence. *see* Sequence
- run sequences, 17
- Run Window, 109
- Sensitivity Analysis, 2, 103
- Sequence, 85, 86, 90, **94–108**, 123, 125, 126
  - Compound, 94, **107–8**
  - DataFile, 94, **99–100**
  - Event, 94, **96–99**, 123
  - MetManager, 94, **101**, 123
  - Nested. *see* Sequence, Compound
  - Parameter, 94, **102–7**
  - Simple, 94
- Shape File, 84, 142
- Simulation Date, 23, 111
- Simulation File, 84, 127
- Soil Moisture. *see*
- Soil Moisture Capacity, 71
- Stage Transfer, 8, 26, 29, 32, 38, 156
- Start State, 33, 35
- Subpopulation, 3, 5, **8–10**, 17, **18**, 21, 25, 31, 35, 41, 56, 60, 73, 74
  - Genetic, **9**, 19, 32, 44
  - Spatial, 10, 19, 31
- Table, i, 3, 18, 109, **120–25**, 125
  - Font Size, 122
  - Format, 122
- Threshold Event, 64
- Time of Day, 23
- Timer, 34, *see* Module type, Timer
- Trigger, 35, 60, 62
  - AtStart, 35
- Unit Update, 106
- Variable, 5, **4–5**, 5, 6, 8, 89, 91, 116, 124, 126, 129, 155
  - Cohort Variable, 26, 30, 35, 42
  - Daily Cycle, 53
  - Delay, 5
  - Demed, 5, 9, 73, 74
  - Summary, 5, 6, 16, 74, 93, 94, 129
- Variable Description, 123