

# Marxan and Relatives: Software for Spatial Conservation Prioritization

Ian R. Ball, Hugh P. Possingham, and Matthew E. Watts

## 14.1 Introduction: The philosophy and history of Marxan

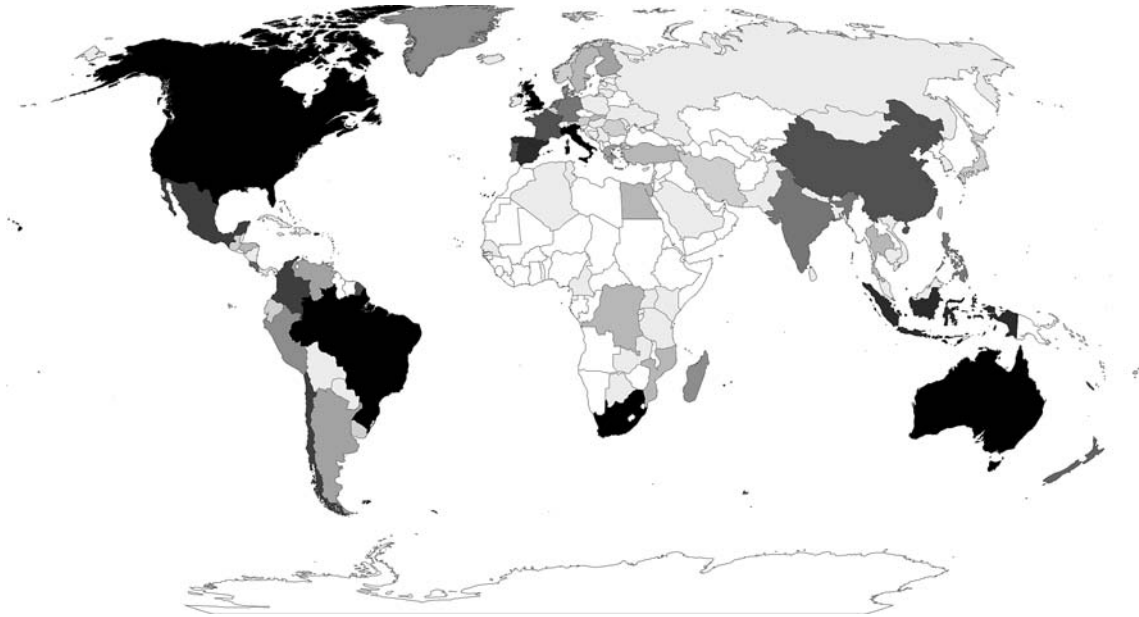
The original intent of Marxan and its predecessors was to solve a version of the minimum set reserve design problem (Cocks and Baird 1989; Chapter 3). In this problem, conservation targets are set for a number of biodiversity features (e.g. three populations of each species), and Marxan selects planning units that represent these targets for a minimum total cost, while allowing for more or less emphasis on spatially clustering the selected planning units. Marxan is now used to solve a range of spatial prioritization problems beyond the selection of reserves. The Marxan software is freely available and interacts with a variety of geographical information system (GIS) tools. It designs clumped reserve systems that make sense to a policy maker or planner.

One of the distinguishing features of Marxan is its use of simulated annealing to find multiple alternative good solutions to the impossibly huge minimum set problem (Kirkpatrick et al. 1983). Why did we choose simulated annealing as the basic optimization algorithm (see Chapter 5 for a discussion of algorithms)? We investigated a number of other methods for solving the minimum set problem including Genetic Algorithms and Branch and Bound methods (Taha 1992). We found that simulated annealing provided good answers quickly and was very flexible. Simulated annealing works well on problems of very different sizes. Non-linearities and other complexities can be added to the problem without having to change the formulation of

the optimization algorithm (Cocklin 1989). In short simulated annealing proved to be relatively fast, relatively simple, and robust to changes in the size and type of the problem.

Marxan has evolved over more than a decade. The philosophy and ideas behind Marxan were developed in the original version of the software, titled Siman. It was written by Ian Ball as part of his PhD thesis under the supervision of Hugh Possingham (Ball 1996) at The University of Adelaide, Australia. SPEXAN, which stands for 'spatially explicit annealing' was the next stage in the development of Marxan (funded by Environment Australia). The Nature Conservancy (TNC) funded, through UC Santa Barbara, a project where Ian linked SPEXAN with ARCVIEW, resulting in SITES. Marxan is a modified version of SPEXAN, partially funded by the Great Barrier Reef Marine Park Authority in Australia and the US National Marine Fisheries Service. Marxan stands for 'marine reserve design using spatially explicit annealing'; however, it is just as applicable to terrestrial conservation planning problems.

The software is changing and developing in a variety of directions. Marxan with zones is the most recent development; it incorporates multiple zones and multiple costs (Watts et al. submitted). The blueprint for Marxan with zones was designed and implemented by the authors of this chapter in collaboration with their colleagues (Watts et al. submitted). Its development was partially supported by Ecotrust and the University of California. Other emerging versions and features of Marxan include:



**Figure 14.1** The world of Marxan: darker colours indicate more Marxan users, and white indicates no Marxan users. Marxan is very widely used and is possibly the most widely used conservation planning software in the world.

- introducing programming efficiencies to improve speed and the capacity to deal with large data sets,
- dealing with probabilistic treatments of threats,
- using cluster analysis to find good, but very different, Marxan solutions, and
- handling asymmetric connectivity for more sophisticated management of connectivity.

One of the most high profile and successful applications of Marxan was the rezoning of the Great Barrier Reef (Fernandes et al. 2005). TNC use Marxan for most of their spatial prioritization work in the USA and around the world. A partial list of Marxan applications – successful, purely academic, and pending – can be found on the Marxan web site ([www.ecology.uq.edu.au/marxan](http://www.ecology.uq.edu.au/marxan)). There are over 1,700 individuals in over 1,200 organizations from more than 100 countries that have used Marxan for marine and terrestrial conservation planning. Figure 14.1 shows the distribution of Marxan users around the world.

Marxan has some strengths as a tool for conservation Planning. First, Marxan uses spatially variable cost data to compute efficient solutions; each planning unit can be assigned a separate cost, which can

be a complex combination of financial, opportunity, and social costs. Second, it uses a powerful optimization technique, simulated annealing, to generate near-optimal solutions. Third, it can generate many solutions quickly, allowing comprehensive data exploration and analysis. Fourth, Marxan solves a well-defined and explicit mathematical problem, as we shall see in the next section. Lastly, because of its widespread use, Marxan is well tested in a range of situations, and training programmes and free online support are provided to make sure that users get the best use out of the software.

Flexibility is probably the key to the success of the package. Due to the simplicity and power of its mathematical formulation, clever problem definition allows it to compute efficient solutions to a range of well-defined conservation planning problems.

## 14.2 What conservation planning problem does Marxan find solutions to?

Marxan finds good solutions to a mathematically well-specified problem which means that there is

no ambiguity about what the software is trying to achieve. The goal in Marxan is to minimize a combination of the cost of the reserve network and the boundary length of the entire system, whilst meeting a set of biodiversity targets. A larger boundary length is considered to be bad for several reasons which include: management costs, increased edge effects, and reduced connectivity. The mathematical problem to which Marxan finds good solutions is:

$$\text{minimize } \sum_i^{N_s} x_i c_i + b \sum_i^{N_s} \sum_h^{N_s} x_i (1 - x_h) cv_{ih} \quad (14.1)$$

subject to the constraint that all the representation targets are met

$$\sum_i^{N_s} x_i r_{ij} \geq T_j \quad \forall j \quad (14.2)$$

and  $x_i$  is either 0 or 1

$$x_i \in \{0,1\} \quad \forall i$$

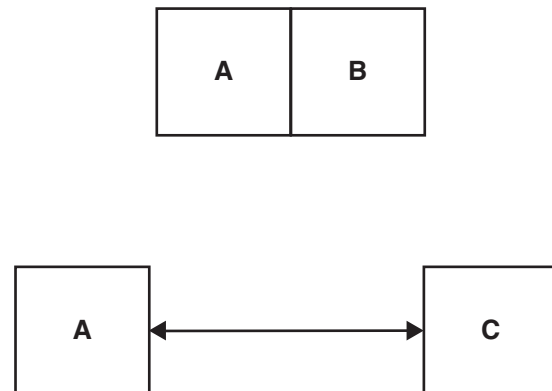
where  $r_{ij}$  is the occurrence level of feature  $j$  in site  $i$ ,  $c_i$  is the cost of site  $i$ ,  $N_s$  is the number of sites,  $N_f$  is the number of features, and  $T_j$  is the target level for feature  $j$ . The control variable  $x_i$  has value 1 for sites selected for the reserve network, and value 0 for sites not selected (see Equation 3.1a).

The first term in Equation 14.1 is a penalty associated with the cost of all the sites (planning units) that are in the system. The second term in Equation 14.1 is a penalty associated with configuration (or shape). A reserve network with a smaller boundary length (or boundary cost) has a more compact configuration. The more fragmented the reserve system, the greater its boundary length. While it is typical for users to set  $cv_{ih}$  to be the length of the physical boundary between sites, and hence favour solutions that reduce the length of open boundaries, it is also possible to use this parameter more innovatively to improve the connectivity of a reserve system. In a more general sense, this second term of Equation 14.1 allows us to introduce a cost (or benefit) for including a particular site and any other site to which that particular site is 'connected'. It is another mechanism that allows the reserve network

to be more than the sum of its parts, in addition to complementarity.

The parameter  $b$  in Equation 14.1 is the boundary multiplier which determines the cost of the reserve system (the first term in Equation 14.1) relative to the penalty for its spatial configuration (the second term in Equation 14.1). The matrix  $CV$  is the connectivity matrix with elements  $cv_{ih}$  which reflects the cost of the connection (such as a boundary) shared by planning units  $i$  and  $h$ . If one site is in the reserve system, and the other is not, then the connection cost must be paid. If both sites are out or in, the connection cost is not paid. In general we encourage users to think about assigning these connectivity costs according to the extra benefits of having any pair of sites in the reserve system. For example,  $cv_{ih}$  could be a quantitative measure of the flow of propagules from sites  $i$  to  $h$  where the sites may be separated by some distance. In this case, Marxan will try to find solutions that maximize the tendency for propagules generated in the reserve network to be retained within the reserve network. The use of a connectivity matrix allows connections between sites that are not adjacent, which means the concept of connectivity within the system can be quite complex (Figure 14.2).

The targets for a given conservation feature,  $T_j$ , can be an amount of that feature which is to be



**Figure 14.2** Two types of connectivity. Sites A and B have a shared boundary. Marxan typically represents this connection as a boundary between the sites. Planning units A and C do not have a shared boundary; however, they are connected due to propagule dispersal between the two sites. Marxan can represent this non-boundary connection in the connectivity matrix.

included within the reserve configuration, or it can be a number of occurrences of that feature within the system. For example we may wish to conserve 30 % of the original extent of every vegetation type or 1,000 adult females of every species. For smaller problems it can also include rules about the contiguity of occurrences of the conservation feature within the system or the separation of representations of the conservation feature within the system ([www.uq.edu.au/marxan/](http://www.uq.edu.au/marxan/)). These software features can be used to ensure a minimum reserve size, or ensure that several representations of each conservation feature are adequately separated in space, the idea of risk spreading. Targets for contiguity and separation features can slowdown the operation of the software substantially if the number of planning units is in the high thousands or greater. If several separate representations of a specific conservation feature are required, e.g. we want a single vegetation type conserved in three separate areas on a continent, it may be best to simply treat that vegetation type as three separate conservation features.

Targets in Marxan are specific to the conservation features and not for other configuration characteristics, such as the minimum size of areas zoned for conservation, or the number of distinct areas zoned for conservation.

Equations 14.1 and 14.2 are the basic formulation for the problem. Marxan solves the problem by placing the objectives (Equation 14.1) and the constraints (Equation 14.2) together into an objective function by transforming the constraints into an additional penalty term. This means that a configuration of planning units which does not meet all of its conservation targets can still be given a value, which is of practical use in the annealing process.

A penalty is given to each conservation feature to either enforce or encourage it to be included in the final configuration to a certain level of representation. If the value of the penalty is less than 1, then a planning unit which only meets that conservation feature's target will tend not to be included, but one which meets other conservation targets at the same time will be included. If the value is greater than 1, Marxan should ensure that the target is met. The penalty is given as a proportion of the target that is not met multiplied by an internally derived penalty value for the conservation feature. This derived

penalty indicates how costly it is to meet the features' target in a reserve specifically for that conservation feature. This is calculated with a greedy algorithm for simple conservation targets and using a variation on the iterative improvement method for more complex target requirements.

### 14.3 How does Marxan work exactly?

A number of different optimization techniques were considered to drive the optimization of the Marxan software. These included the exact optimization method of integer linear programming as well as approximating optimization methods such as genetic algorithms and simulated annealing (see Chapters 4 and 5). The serial selection heuristics which were traditionally applied to reserve selection, including the useful greedy algorithm, were also tested.

Integer programming can guarantee an optimal solution to a problem of the form given in Equations 14.1 and 14.2, and some have argued that these classical approaches are best for conservation planning problems (Underhill 1994; Rodrigues and Gaston 2002). However, we found two drawbacks of these more rigorous classical methods – first and foremost they failed to solve extremely large problems; and second, for practical and political reasons, finding the single best solution is not that useful in conservation planning.

The problem described in Equations 14.1 and 14.2 and those in Chapter 3 are in the mathematical category of NP-Complete. Optimal solution times for problems of this category rise faster than linearly with the size of the problem, where the problem size is the number of planning units and number of conservation features (each of which creates a constraint). Our initial explorations had long solution times for problems with 2,000 planning units and 250 conservation features and simple planning requirements (i.e. no spatial configuration requirements, Pressey et al. 1997). Integer programming was considered too slow to deal with problems of increasing complexity and size, where the spatial configuration is important and uncertainty in data and parameters meant there was a need for extensive sensitivity analysis. Increases in computational speed continue, but larger non-linear conservation

planning problems are still insoluble in reasonable time using classical operations research methods.

A criticism which is occasionally directed at the configurations produced by Marxan is that none of them are necessarily the optimal solution. However, it was considered more important to be able to rapidly produce a range of solutions which are near optimal to enable decision makers to negotiate and make choices amongst a good range of options. The data inputs into the problem are often imprecise, including estimates of the abundance of conservation features, or their occurrence pattern, the costs associated with planning unit, and the cost associated with the connectivity values between each pair of planning units. Also there is often an arbitrariness associated with each of the conservation targets. This means finding the single best system is somewhat academic (Pressey et al. 1996b). If possible it can be instructive to see how suboptimal Marxan solutions are for problems which can be solved using classical methods (Fischer and Church 2005).

Simulated Annealing is described in detail in Chapter 5. In order for it to work, one needs an objective function which can be evaluated explicitly for any valid configuration of planning units. By combining the constraint of Equation 14.2 as a penalty factor appearing in Equation 14.1, it is possible to give a value to any configuration of planning units whether they satisfy all the conservation targets or not. This is not just a convenience for the application of the optimization algorithm, but it is generally useful in allowing targets which are unreachable, or targets which only need to be partially fulfilled, or in evaluating the state of alternative configurations which do not meet all the targets.

In our implementation of simulated annealing, an initial potential solution is created either from a user-defined starting point or by randomly selecting some proportion of planning units (which might be all or none of them). New trial solutions are generated iteratively by randomly changing the status of a single planning unit and assessing the new configuration in terms of an improved or worsened objective function value. For example, if the planning unit was not previously part of the proposed configuration and its random inclusion improves the system, then it is kept in, if not it is

removed. Similarly, if it was in the original solution and its random exclusion improves the system it is excluded, if not it remains included.

The heuristics that we tested tended to produce inferior results. Also, they produce only a single result, although they run quickly and are useful for a rough exploration of different scenarios. However, some of the Greedy Algorithms (described in Chapter 5) can also be robust and may prove a suitable alternative for some problems, especially those without complex spatial constraints (Pressey et al. 1997). Hence, they can be run within Marxan.

#### 14.4 The Marxan user interface and output

Marxan is a command line program written in the C programming language. The version available for download is compiled and tested on the Microsoft Windows operating system, although it can also be compiled for other platforms (e.g. Linux). The data input files for Marxan are simply constructed text files which can be edited with any text editor. The program supports a number of types of simple formatting such as comma-delimited values or tab-delimited values which make it easier to edit in programs such as Microsoft Excel, OpenOffice.org Calc, Arc-View, S, or R. It can be used as a stand-alone program or as a plug-in for another decision support system (DSS) such as C-Plan, CLUZ, PANDA, or Vista. These DSS provide graphical outputs (maps and charts) and facilitate interpretation and manipulation of Marxan inputs and outputs. Hence they considerably enhance the ability of Marxan to help managers and stakeholders make decisions.

There are two standard Marxan outputs. First, the best solution file lists the reserve network with the lowest score from all the good reserve networks generated. Second, the summed solution file records the selection frequency for sites across all the good reserve networks generated.

In addition to being used to generate a single configuration, or a number of alternative configurations, the software can be used to produce nested configurations by iteratively running it with increasing representation targets. In this way it can produce core areas and then partially protected areas built around those core areas, a kind of zoning.

The selection frequency of a planning unit is a measure of how important it is to get that planning unit to meet all the targets. It is calculated by the fraction of the alternative solutions, derived from the same data and targets, in which a planning unit is selected. The selection frequency map indicates which areas are more often included in configurations and which are not. This is often used as a surrogate for irreplaceability (Ferrier et al. 2000). The planning units are infrequently selected if there are a range of equally good alternatives. If they are truly irreplaceable, then they will appear in every solution. Of particular interest are those planning units which are frequently included but have a high cost associated with them. Highlighting these can focus the attention of the user on areas where hard decisions need to be made.

Marxan has been constructed for use within a range of different graphical front-ends. First, GIS software changes a lot and different people use different packages. For example CLUZ is a front end for Marxan that runs on ArcView 3.X, a widely used commercial GIS package that will eventually be replaced by other GIS software (Smith 2004). Second, this method encourages other programmers to develop front ends for Marxan that can be tailored to the needs of their organization.

Detailed information about how to use Marxan can be found in a new user manual (Game and Grantham 2008) while solid advice on tricks and pitfalls can be found in the new Marxan Good Practices Handbook (<http://www.uq.edu.au/marxan/>).

### 14.5 The Great Barrier Reef: An example of how Marxan has been used

Marxan was used as one of the decision support tools for rezoning the Great Barrier Reef, Australia. The Great Barrier Reef was rezoned on July 1 2004, and it represents the largest successful, and most complex, real-world application of systematic conservation planning principles (Fernandes et al. 2005). The rezoning took the Great Barrier Reef Marine Park Authority (GBRMPA) a decade to complete, highlighting the social and political complexities of conservation planning at this huge scale. Here, we give a very brief overview of how Marxan was used, the kinds of principles and data

that underpinned its application, and some of the pros and cons of its application in this situation.

The Great Barrier Reef is a huge area off the north-east coast of Australia. It is a multi-use marine park that before 2004 had less than 5 % of its extent in no-take (no-fishing) zones. The rezoning raised the percentage that was in no-take zones to over 33 %. The target for each feature type was 20 %, and this was met for all but a couple of feature types. The use of Marxan for this exercise was interesting for a number of ecological and technical reasons:

- The region was divided into over 17,000 planning units where each reef, and a buffer around each reef, was a planning unit. The area between reefs was divided into hexagons of around 12 km<sup>2</sup>.
- The cost of each planning units was derived from a linear combination of commercial fishing interests, recreational fishing interests, indigenous interests, and interests of people who would prefer particular places to be inside the reserve system (a negative cost).
- The most important biodiversity feature layer was the 'bioregions'. The whole system was divided into 70 bioregional types. These bioregions were derived from extensive statistical modelling and expert analysis of ecological and biophysical data. Targets of 20 %, and higher for rare reef types, were set for each feature.
- Reserves were preferentially set to be bigger than 400 km<sup>2</sup> where possible and each bioregion was meant to be represented in at least three well-separated reserves of at least this size. The minimum reserve size and minimum separation facilities of Marxan were used to achieve this.

It is instructive to note that GBRMPA initially ignored social and economic information and tried to get Marxan to find a biologically 'pure' conservation plan, a plan with only biological information. This is frequently done, or aspired to, but is fraught with problems. First, area is usually assumed to be cost, so that the problems of economic and social considerations were ignored, and the area conserved was simply minimized while meeting all conservation targets. This failed to deliver a system that was socially and economically viable.

Second, Marxan can be very indecisive if there are many planning units with roughly equal costs and equal amounts of different conservation features. It produced selection frequency outputs where most planning units have a selection frequency of around 20 % (the target set by GBRMPA) which is not very useful as decision support for planning.

While Marxan contributed to the selection of no-take areas, the final zoning places each part of the region into one of seven zones. Our new software, Marxan with zones, would be ideally suited to similar problems of marine park zoning.

#### 14.6 Marxan applied to the Australian continent

Marxan has been used to do a conservation prioritization for the entire Australian continent (Carwardine et al. 2008; Klein et al. 2009 in press). This application of Marxan involved a few innovations that we would like to highlight.

First, the planning units used for the whole continent were sub-catchments (Stein 2005, 2006). By using sub-catchments ( $n = 62,630$ ) and the knowledge of which sub-catchments were connected by the flow of water (e.g. the share the same river or stream), priority reserve networks were preferentially grouped along waterways, as illustrated in Figure 14.3. This has been perceived to be a need for many people interested in freshwater conservation planning and incorporating ecological processes into conservation planning (Chapters 2 and 7).

Second, a large number and array of conservation features were included:

- 1,763 vegetation types, identified by intersecting 62 native vegetation subgroups (The National Vegetation Information System 2001) with 85 bioregions (Australian Government Version 6.1),
- 563 bird species distributions modelled using Birds Australia location data (Carwardine et al. 2006),
- 1,222 threatened species distributions (Commonwealth of Australia 1999), and
- 151 environmental domains, represented by an environmental classification based on a set of key

climatic, topographic, and substrate conditions that characterize the landscape (Mackey et al. 2008).

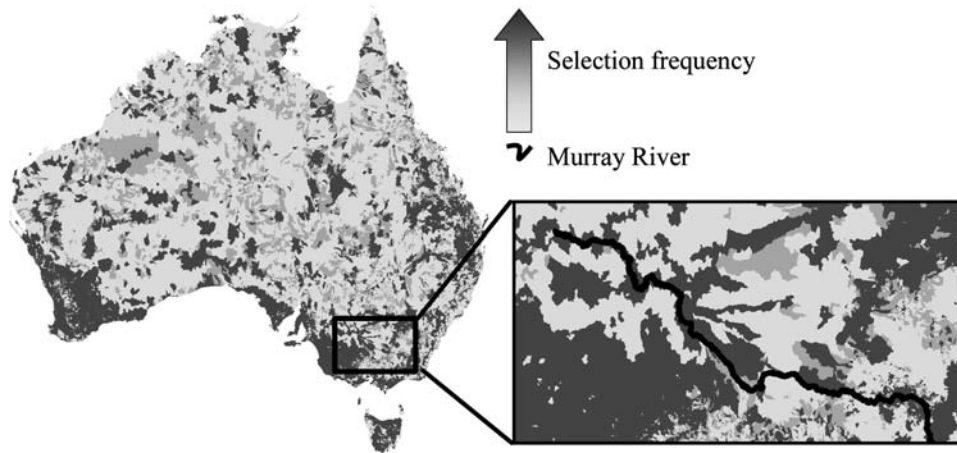
Third, with the aim of protecting a selection of ecological and evolutionary processes that maintain and sustain biodiversity, two types of refugia were targeted:

- Drought refugia: Areas which are critical for the persistence of many species during harsh climatic conditions in arid Australia. We identified drought refugia, areas of high and regular herbage production in arid and semi-arid Australia, using estimates of gross primary productivity modelled from high-resolution satellite data, and spatially interpolated climate data (Figure 2.3).
- Evolutionary refugia: Areas which are important for maintaining and generating unique biota during long-term climatic changes. Evolutionary refugia were identified by experts as areas in Australia important for generating species (Morton et al. 1995).

Fourth, areas were prioritized for two different actions, land acquisition for the national reserve system and stewardship arrangements on private land. Many users think that software like Marxan is just about building a network of protected areas. However, if we assume the cost of a planning unit is the cost of entering into a stewardship or other conservation arrangement on private land, then it can be used to prioritize places for these sorts of conservation action.

Finally large intact areas in good condition were preferentially targeted rather than fragmented landscapes (Klein et al. 2009 in press). Using data on the condition (wilderness quality) of each sub-catchment, we either choose to meet targets preferentially in planning units in good condition or choose large clusters where adjacent sites in good condition were well connected.

The aim of this study was to guide the Australian government in identifying spatially explicit priorities for biodiversity conservation using fundamental concepts of spatial conservation prioritization (Chapter 2) at a continental scale. An example of a final prioritization, using a target of 30 % of each feature and with a bias towards large intact landscapes is shown in Figure 14.3.



**Figure 14.3** A majority of the sub-catchments along the Murray River were prioritized for conservation investment when the connectivity of sub-catchments was considered. Darker colours indicate a higher selection frequency or priority for investment.

### 14.7 Applicability and limitations of Marxan

Marxan runs from a command line interface and therefore has no graphical user interface of its own. This is a strength of Marxan as its architecture allows use by any DSS as a plug-in, and it does not become outdated from being tied to a particular DSS, GIS, or operating system.

There is inherent complexity in setting up input files and interpreting output files, which are both a strength and a weakness. It is a strength as it requires users to think through their problem definition in detail to manage these complexities, so the user is more likely to understand the problem they are solving. It is a weakness as the management of files is not a simple point and click exercise, leading to many users becoming frustrated.

Indeed most problems using Marxan arise when input files do not conform to the standard data definition. These problems can be avoided by careful application of the input-file specifications laid out in the Marxan user, or by using one of the many DSSs that support the building of Marxan input files. Improved error checking and data diagnostics would reduce this limitation of the software.

Marxan does not readily deal with the thorny issue of conservation adequacy. Probably the most sensible approach is to set targets for key species using

outputs from viability analyses (see Chapters 8 and 9). In more recent applications surrogates for conservation adequacy have been used where ecosystem processes, vegetation condition, and evolutionary refugia are included in the formulation of the problem.

There is the challenge of including even more complex problems like ecological processes, social dynamics, and continuous benefit functions (Chapter 2). The desire to include more functionality and improve the software should always be tempered by consideration of whether important complexities are better handled within Marxan, or outside of Marxan in a DSS. This is particularly true when we consider constructing the cost of each planning unit—a parameter that may be built up from a great deal of other information as in the Great Barrier Reef rezoning case study.

Marxan is designed to be open-ended with regard to the size of the problems to which it can be applied. Users have successfully run problems with over 2,000,000 planning units. The software is designed to be efficient in memory usage so that the number of conservation feature records is one of the key memory components. A sparse matrix of occurrences of conservation features can require little memory even if there are very large numbers of planning units or conservation features. Typically the factor which slows the software the most is the number of planning

units – there tends to be a limit to the number of conservation features which can be usefully considered, but the number of planning units can always be increased by finer subdivisions of space. The ideal number of iterations to use in the system is largely a function of the number of planning units so that increasing them increases both the memory requirements and running time of the software.

### 14.8 Common misconceptions and misapplications of Marxan

In this section, we describe some of the misconceptions about Marxan and how it can be applied inappropriately. Some of these misunderstandings are common to all conservation planning software.

First we discuss the ‘black box’ issue. Some users ask questions about why different planning units were chosen and how the algorithm goes about selecting planning units for inclusion into the configuration. Because Marxan does not serially select planning units the way a designer might do it by hand, it appears that the selection mechanism is a mysterious ‘black box’.

This concern arises partially because of a lack of understanding of how the simulated annealing algorithm works, and mathematical optimization in general. Its operations are viewed as complex and mysterious, and many users feel uncomfortable with the technical description of the algorithm as given in the manual and supporting papers. There are a variety of descriptions of how simulated annealing works associated with Marxan and CLUZ, including an online game.

The other problem comes from focussing on the wrong part of the system. Historically the focus of systematic planning was on the process of serially selecting planning units to build a reserve network and software was designed to replicate the manual serial selection of planning units. However, the key to understanding the system is to understand the problem formulation: the objective function and the choices which are made in terms of setting targets, differentially weighting different conservation features, planning units, connections or boundary lengths, and the relative weightings of the cost term and the connectivity term. If we understand the objective function that is being optimized, then we know what

problem is being solved (see Section 14.2). The optimization method can then well be a black box without the user being disadvantaged; in the same way that the user of a calculator need not know the internal method by which multiplications are computed. Indeed the process of serially selecting planning units can be quite misleading, because it assumes myopic algorithms are useful, and it tends to ignore the fact that the whole reserve system has a value, not individual planning units – in reserve system design the whole is more than the sum of the parts.

It is still useful for the user to understand the relative strength and weakness of the optimization methods (covered in Chapters 4 and 5) as well as having a practical understanding of setting different values for the parameters that control the optimization method.

Some users will only apply the in-built heuristics such as the richness, or greedy, algorithm. This can produce quick results and is a fast and useful way of exploring trial solutions. Because it is implemented as a greedy algorithm, it can both add and remove planning units from a configuration. However, even though these heuristics, and a number of the other serial heuristics, are included in Marxan, it is a mistake to use these as a complete substitute for simulated annealing. The serial heuristics produce solutions which are inferior to the simulated annealing method with the greedy algorithm for example being driven by the distribution of the planning units with the highest density of conservation feature representation. An ordering of planning unit selection and deletion is produced which can be useful in the planning process, but it misses out on the power of simulated annealing to find solutions to the more complex and interesting non-linear versions of this problem.

The production of alternative configurations of conservation areas, particularly through the selection frequency measure, is one of the strengths of Marxan, but some have complained that this is an indication that the solutions are not optimum. Although the authors would recommend against it, the user has the option of saving only the best configuration from an arbitrary number of repeats or of producing only a single solution using a much longer annealing trial. At the end, the main thing is that it produces significantly better solutions than could be

obtained without using a computational algorithm (Klein et al. 2008b; Leathwick et al. 2008c).

One use of Marxan on which there is divided opinion is in the creation of cost layers. One way to set the cost associated with each planning unit in the configuration is to combine the costs from a number of different sources into a single value by differentially weighting these sources. For example there might be abstract social costs for including planning units in the network, actual setup costs, and opportunity costs to a variety of users of the system such as recreational and commercial fishermen. If these are combined into a single cost then a weighting for each type of cost must be determined, and it is not always clear how this weighting should be determined in an objective manner.

The alternative, which some of the developers recommend, is to produce alternative configurations each using a separate cost layer. This can help focus attention in the design process on those areas which are most different in their impact on different user groups.

Another complaint that is often directed at Marxan is that if the problem definition is not correct then invalid results will be obtained. This is, of course, as obvious as it is important when considering the results from Marxan or any other systematic planning software. The alternative is to use some informal method in which the problem definition is never made explicit, which in our view is poor planning. Good problem definition lies at the heart of good conservation (Possingham et al. 2001).

## 14.9 The future of Marxan

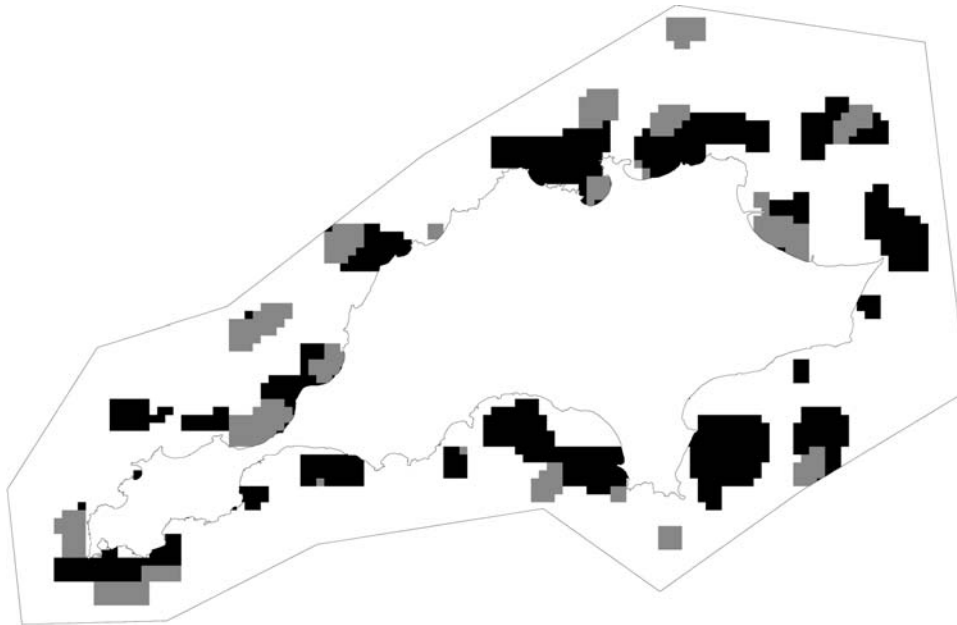
Marxan is continually changing as we work with partners to provide new functions and more flexibility. In this section, we briefly introduce three of the most substantive new functionalities for Marxan.

'Marxan with Zones' is a decision support tool developed under contract to Ecotrust. It was developed to support the design of marine protected areas along California's coast as part of California's Marine Life Protection Act (Klein et al. 2008b). Marxan with Zones is an extension of the Marxan software, developed by Ian Ball, Matthew Watts, and Hugh Possingham. It has the same functionality as Marxan but is also able to set priorities for multiple zones (i.e. marine protected areas of various protection

levels) and incorporate multiple costs into a systematic design framework. The purpose of Marxan with Zones is to assign each planning unit in a study region to a particular zone while meeting a number of biodiversity targets at a minimum total cost. This software clearly has many advantages and is ideally suited to many current marine zoning problems as well as terrestrial problems where more than one land-use change is under consideration. A preliminary output from Marxan with zones applied to the area around Rottnef Island is shown in Figure 14.4.

We will not discuss Marxan with Zones in detail here but emphasize two important considerations. First, this software requires a lot of additional data – for example one must specify the cost of placing a planning unit into any one of the different zones, the benefit to each biodiversity feature of being placed into a particular zone, and the relative merits of having each zone type juxtaposed with another zone type. Figure 14.1 shows that clever use of the latter concept enables us to create a zoning map where highly protected areas can be buffered by less protected areas. Second, Marxan with Zones has a strong focus on the interaction between different zones. Instead of a simple boundary between planning units which are in the configuration and those which are not, every planning unit can be in one of many different types of zones, some of which have a variety of conservation uses and others which are set aside for specifically non-conservation uses. The boundaries or connectivity between each planning unit is even more important than the simpler case and the matrix of different types of zone connectivity adds more control over the system as well as more complexity.

In Section 3.5, we discussed the fact that we are often unsure exactly whether or not a feature is present in any particular planning unit, and if so, how much of it is there. This uncertainty may arise from the habitat modelling package or population survey used to create feature distribution maps or the possibility that the feature has or may appear or disappear from the planning unit (through habitat change, catastrophes, or population dynamics). Indeed, there are many reasons why we may wish to assume that the representation matrix (Chapter 3) is filled with probabilities and not absolute numbers. A new version of Marxan, currently undergoing testing, will be able to solve problems where there is uncertainty about feature presence. In particular, it



**Figure 14.4** A zoning plan of Rottneest Island (Western Australia) created with Marxan with Zones showing two kinds of zone. The black zone is a proposed no access zone, grey is a proposed recreation zone with no-fishing, and white is a proposed recreational fishing zone. You can see how the grey zone tends to buffer the black zone, creating a recreational no-fishing zone around the no-access zone. This is generated by placing a high cost on boundaries that connect the most incompatible zones – no-access and recreational fishing.

will allow users to set not just a feature target,  $T_f$ , but also the confidence with which we want to meet that target. For example, we may wish to be 90 % sure that there are 2,000 ha of seagrass bed in our marine reserve. This idea is to build reserve networks that are robust to uncertainties pertaining to catastrophic events, and changes in feature distribution through disturbance and succession (Game et al. 2008; Drechsler et al. 2009 in press).

We have implemented an automated cluster analysis of solutions to explore the many different solutions generated by Marxan. Typically, the best solution (which is simply the solution with the lowest cost) and the summed solution (the selection frequency of planning units over a number of solutions) are the only Marxan outputs used. We use the R statistics package to perform hierarchical cluster analysis and hence a selection of good but different solutions. This is particularly useful where planners want a range of credible options to consider.

New front ends are being developed for Marxan and new functionality is being added to existing front ends to make them easier to use. A version of Marxan that runs natively in the Linux operating

system has recently been developed and tested. We encourage other researchers to cooperate with us in the development of this free tool – its development has always been very much a cooperative and international venture.

### Acknowledgements

Marxan has developed over many years with advice and input from many people – too many to name. However we would like to particularly thank: Carissa Klein, Josie Carwardine, Jeff Ardron, Bob Smith, Natalie Ban, Doug Fischer, Sandy Andelman, Kerrie Wilson, Romola Stewart, Lindsay Kirchner, and Dan Segan.

Marxan would not have existed without some financial support. While it is free software, and its support is funded through the University of Queensland and the spare time of generous individuals – several groups have paid for important developments and these include: the Environment Department (Australia), the Great Barrier Reef Marine Park Authority, Ecotrust (USA), the Nature Conservancy, and the US National Marine Fisheries Service.