rapr: Representative and Adequate Prioritisations in R

Jeffrey O. Hanson[1], Jonathan R. Rhodes[2], Hugh P. Possingham[1], Richard A. Fuller[1]
[1]School of Biological Sciences, The University of Queensland, Brisbane, QLD, Australia
[2]School of Geography, Planning and Environmental Management, The University of Queensland, Brisbane, QLD, Australia
Correspondance should be addressed to jeffrey.hanson@uqconnect.edu.au

09 March 2016

Abstract

A central aim in conservation is to maximise the long-term persistence of biodiversity. To fulfil this aim, reserve networks are used to safeguard biodiversity patterns (eg. species, populations) and processes (eg. evolutionary processes that underpin genetic variation). Reserve selection is often formulated as an optimisation problem to identify cost-effective prioritisations. However, most existing decision support tools are based on formulations that are well suited for preserving biodiversity patterns, but not biodiversity processes. To fill this gap in the conservation planning toolbox, we developed the exttt{rapr R} package. This exttt{R} package provides a toolkit to guide reserve selection using novel formulations of this problem. Here, we explore the functionality of this R package using simulated species and a conservation planning exercise in Queensland, Australia as a case-study. We demonstrate how explicitly considering biodiversity processes can alter a prioritisation. In most cases, we found that only a few additional plannning units are required to sufficiently preserve of biodiversity processes.

Short running title: Representative and adequate prioritisations
Word count: XXXXX
Number of references: XXXX
Number of figures, table and text boxes: 4

Contents

Introduction

The overarching aim of conservation is to maximise the long-term persistence of biodiversity (McNeely 1994; Margules and Pressey 2000). To achieve this, conservation actions must preserve biodiversity patterns (eg. species, populations) and the processes that sustain them. One of the major tangible achievements of modern conservation has been the act of setting aside areas for preservation (Sanderson *et al.* 2015). Reserve networks buffer species from threatening processes (eg. urbanisation; Margules and Pressey 2000) and set the stage for direct management interventions (eg. captive breeding and reintroduction programs; Kleiman 1989). However, the resources available for conservation action are limited, and so reserve networks must be sited in places that satisfy conservation objectives for minimum cost (Margules and Pressey 2000). To achieve this, reserve selection is often formulated as an optimisation problem and then solved to identify cost-effective candidate reserve systems (prioritisations; Margules and Pressey 2000).

To fulfil the overarching aims of conservation, reserve networks must preserve both ecological and evolutionary processes (Crandall *et al.* 2000; Margules and Pressey 2000). Ecological processes, such as predator-prey interactions, pollination, and decomposition, are required for biodiversity to persist over short time-scales. Typically, they operate over small geographic domains–with exceptions such as migration and refugial habitats–and can be preserved using suitably large planning units (Ciarleglio *et al.* 2009) that each contain a discrete unit of habitat (Klein *et al.* 2009). On the other hand, evolutionary processes are required for biodiversity to persist over long time-scales, and they typically operate over large geographic domains. Protected areas must preserve adaptive evolutionary processes to foster resilience against environmental change [eg. climate change]. Protected areas must also preserve neutral evolutionary processes. These processes are associated with the breakdown of gene flow between populations. They are important for maintaining genetic diversity, and avoiding inbreeding depression (Moritz 1999). In recent decades, a wealth of data on biodiversity processes has become available to conservation planners [eg. bioclimatic, genetic, and trait data; Hijmans et al. (2005); Vos et al. (1995); @481]. Yet this data is only rarely used to guide conservation planning (Hendry *et al.* 2010). This is due to that fact that existing reserve selection tools focus on preserving biodiversity patterns or processes–but not both.

Many decision support tools have been developed to help identify cost-effective prioritisations (eg. `ConsNet`, Ciarleglio *et al.* 2009; `Marxan`, Ball *et al.* 2009; `Zonation`, Moilanen 2007). Typically, these tools are used to deliver a prioritisation that preserves multiple species. Given a set of target species (or features), decision makers can generate prioritisations that preserve biodiversity patterns by ensuring that an adequate proportion of each species' range is secured. These tools, however, are not well-suited for identifying prioritisations that secure biodiversity processes (but see Faith 2003). To preserve biodiversity processes, a prioritisation must preserve a representative sample of each species. For instance, to preserve predator-prey interactions, a prioritisation must preserve individuals from each predator and prey species in the same area. To preserve neutral evolutionary processes, a prioritisation must secure individuals descended from each of the genetic lineages that comprise each species considered in the exercise (Moritz 1994). To preserve adaptive evolutionary processes, a prioritisation must preserve the adaptive landscape of each species–it must preserve individuals experiencing different selection pressures (Moritz 2002). To accommodate such information into multi-species, species-oriented reserve selection tools (such as `Marxan` and `Zonation`), species ranges can be partitioned into different groups using additional data as a pre-processing step (eg. based on habitat type; Carvalho *et al.* 2011). Each partition can then be treated as a distinct feature when generating prioritisations. However, this approach is limited because it relies on categorical data. Data on biodiversity processes is often continuous and hyper-dimensional, and often cannot be reduced to a few categories without significant information loss (Faith and Walker 1996).

Today, one of the key issues in reserve selection is the lack of a unifying decision support tool that can accommodate non-discrete data on biodiversity processes in a multi-species context. To fill this void, we present the `rapr` `R` package. This `R` package uses novel formulations of the reserve selection problem to provide decision makers with the tools to generate prioritisations that preserve biodiversity patterns and processes. We also provide a tutorial showcasing the functionality of this `R` package using simulated species and a case-study conservation planning scenario in Queensland, Australia.

Problem formulations

The `rapr` `R` package uses two novel formulations of the reserve selection problem to identify cost-effective prioritisations. These formulations share many constraints and variables. For brevity, the variables used by both formulations will be defined. Biodiversity features are defined as the entity(s) that the prioritisation is required to preserve (eg. species, populations). Spatial attributes are defined as the intra-feature variation that the prioritisation is required to sample. These attributes are related to the biodiversity processes that the prioritisation needs to represent (eg. environmental variation, and genetic variation).

Each attribute is conceptualised as a space. This space is termed an attribute space. Each planning unit is thought to occupy a single point inside each space. For example, a decision maker may require a prioritisation that preserves populations along climatic gradients. To achieve this, the decision maker might use an "climatic" attribute space with dimensions relating to mean annual temperature (°C) and precipitation (mm). Any given combination of temperature and precipitation may be conceived as a point in this environmental space. By associating planning units with climatic data, they can be mapped from geographic space to this environmental attribute space.

Demand points are points that also exist in an attribute space. They are designated by the decision maker to indicate regions of the attribute space that should be preserved in the prioritisation (see below for discussion on how demand points can be generated for real-world datasets). The amount of variation in the attribute space that a prioritisation secures is a function of the distance between each demand point and each selected planning unit in the attribute space. The shorter the distances between the demand points and the planning units; the better the prioritisation is at securing the variation in the spatial attribute. To convert these amounts to a proportion–a meaningful unit for a decision maker–the distances between the selected planing units and the demand points are scaled by the distances between the demand points to the centroid of the demand points. In any attribute space there may exist points that are impossible (eg. mean annual rainfall -5 mm), do not occur in the study area (eg. mean annual temperature 30°C in Antarctica). Additionally, there may be some regions that are desirable for some features and undesirable for others (eg. conditions known to be outside the physiological tolerance of a species). Thus a different set of demand points and weights are used for each attribute space and each feature. By placing demand points in desirable regions of an attribute space for a given feature, the decision maker can ensure that prioritisations secure the feature in planning units with spatial attributes that are desirable for that feature.

To illustrate these concepts, we will briefly describe a conservation planning scenario example involving attribute spaces and demand points. A decision maker may wish to develop a prioritisation for a single species. This species has four populations in the study area. These populations are in the process of divergent evolution, with different populations inhabiting different environmental conditions and accruing different adaptations. However, the decision maker can only afford to preserve three of the populations. The decision maker needs to select a set of populations that will secure the most of intra-specific variation. To describe this intra-specific variation–given that no genetic data was available–the decision maker obtained data on the environmental conditions (rainfall (ml) and temperature (°C)) where each population was found. The decision maker then used this environmental data to construct a two-dimensional environmental attribute space. Next, the decision maker generated demand points as equi-distant points between the range of values where the populations were found (± 20% to avoid edge effects; Faith and Walker 1996). By comparing the distribution of the demand points to the distribution of the populations in the attribute space, the decision maker can identify a suitable prioritisation (Figure 1). We can see that if the decision maker preserves both populations A and C, they will effectively "double-up" on the same environmental characteristics, and in turn their waste resources. Instead, a more representative sample of the intra-specific variation could be preserved by securing populations A, B, and D. This example demonstrates how the inclusion of biodiversity processes can guide the reserve selection process.

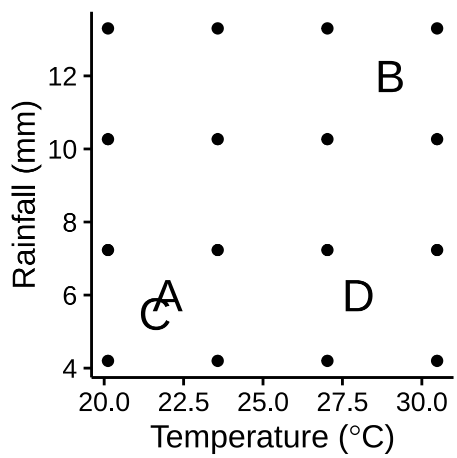


Figure 1 Example of an attribute space. This environmental attribute space has dimensions relating to annual temperature (°C) and rainfall (ml) values. Letters denote the environmental conditions associated with the geographic locations where four hypothetical populations are found. Points represent demand points. In this space, populations closer to each other are considered more similar to each other.

The problem formulations used to guide reserve selection in the `rapr` `R` package are based on a combination of the `Marxan` reserve selection problem and the uncapacitated facility location problems (Cornuéjols *et al.* 1990). All mathematical terms defined hereafter are described in Table S1 for convenience. For convenience, the cardinality of sets will be denoted using the same symbol used to denote the variable. Define F to be the set of features (indexed by f). Let J be a set of planning units (indexed by j). Also, let A_j denote the area, and C_j denote the cost of preserving planning unit $j \in J$. To assess the extent to which each feature is secured in a given prioritisation, let q_{fj} denote the probability of feature f occupying planning unit j. The level of fragmentation associated with a prioritisation is parametrised as the net exposed boundary length. Let the shared edges between each planning unit $j \in J$ and $k \in J$ be e_{jk}.

Let S denote a set of attribute spaces (indexed by s). Each $j \in J$ is associated with spatially explicit data that represent coordinates for each attribute space $s \in S$. Let I_{fsi} denote a set of demand points (indexed by i) for each feature $f \in F$ and each attribute space $s \in S$. Let λ_{fsi} denote the weighting for each demand point $i \in I$, $f \in F$ and $s \in S$. Let d_{fsij} denote the distance between each demand point $i \in I$ and each planning unit $j \in J$ for each feature $f \in F$ and attribute space $s \in S$. To describe the inherent variation in the distribution of demand points for feature f and space s, let δ_{fsi} denote the distance between each demand point $i \in I$ and the centroid of the demand points. Demand points with greater weight λ_{fsi} are more important, and the optimal solution will be likely to select planning units close to highly weighted demand points. As a consequence, the decision maker will need to choose an appropriate weighting for each demand point. The decision maker will also need to choose an appropriate distance metric for each attribute space. For example, Euclidean, Mahalanobis (Mahalanobis 1936), Bray-Curtis, or other distance metrics may be appropriate given the nature of the attribute space (evaluated in Faith *et al.* 1987).

Targets are used to ensure that prioritisations adequately preserve each species. Amount-based targets are used to ensure that the total amount of habitat preserved is sufficient. Let T_f denote the expected amount of area that needs to be preserved for each feature $f \in F$. Space-based targets ensure that a sufficient proportion of the intra-specific variation is secured. Let τ_{fs} denote the space-based targets for feature $f \in F$ and attribute space $a \in A$. For convenience, these both types of targets are expressed as proportions in the `R` package.

The `R` package provides two formulations for reserve selection. They are based on the unreliable (Cornuéjols *et al.* 1990) and reliable uncapacitated facility location (Cui *et al.* 2010) problems. The key difference between them is that the reliable formulation explicitly considers the probability that the planning units are occupied when calculating the proportion of variation that a given solution secures, whereas the unreliable formulation does not.

Unreliable formulation

In the unreliable formulation, the control variables are the B, T_s, and τ_{sa} variables.

$$
\begin{aligned}
T_s &= \text{amount target for feature } f && \text{(1a)}\\
\tau_{sa} &= \text{representation target for feature } f \text{ in attribute space } a && \text{(1b)}\\
B &= \text{boundary length modifier (BLM): penalise fragmented solutions} && \text{(1c)}
\end{aligned}
$$

The decision variables are the X_j and Y_{fsij} variables.

$$
X_j = \begin{cases} 1, & \text{if planning unit } j \text{ is selected for conservation action} \\ 0, & \text{otherwise} \end{cases} \tag{2a}
$$

$$
Y_{fsij} = \begin{cases} 1, & \begin{array}{l}\text{if demand point } i \text{ is assigned to planning unit } j \text{ for feature} \\ f \text{ in space } s.\end{array} \\ 0, & \text{otherwise} \end{cases} \tag{2b}
$$

Each demand point $i \in I$ for feature $f \in F$ and space $s \in S$ is assigned to a selected planning unit J where $X_j = 1$. The weighted distance between the demand point and its assigned planning unit

$\lambda_{fsi} d_{fsij}$ is used to assess how well the demand point is represented in a given solution. Generally, demand points are assigned to the closest selected planning units (unless particularly low space-based targets are used).

The unreliable formulation (URAP) is a defined as a multi-objective optimisation problem.

$$
\begin{aligned}
&\text{(URAP)} \qquad \text{Min} \sum_{j=0}^{J-1} (X_j C_j) + \sum_{j=0}^{J-1} \sum_{k=j}^{J-1} X_j (1 - X_k)(B e_{jk}) + && && \text{(3a)} \\
&\text{s.t.} \sum_{j=0}^{J-1} A_j q_{fj} \geq T_f && \forall 0 \leq f \leq F-1 && \text{(3b)} \\
&1 - \frac{\sum_{i=0}^{I-1} \sum_{j=0}^{J-1} (\lambda_{fsi} d_{fsij} Y_{fsij})^2}{\sum_{i=0}^{I-1} (\lambda_{fsi} \delta_{fsi})^2} \geq \tau_{fs} && \forall 0 \leq f \leq F-1, && \text{(3c)} \\
& && 0 \leq s \leq S-1 && \\
&\sum_{j=0}^{J-1} Y_{fsij} = 1 && \forall 0 \leq f \leq F-1, && \text{(3d)} \\
& && 0 \leq s \leq S-1, && \\
& && 0 \leq i \leq I-1 && \\
&Y_{fsij} \leq X_j && \forall 0 \leq f \leq F-1, && \text{(3e)} \\
& && 0 \leq s \leq S-1, && \\
& && 0 \leq i \leq I-1, && \\
& && 0 \leq j \leq J-1 && \\
&X_j, Y_{fsij} \in 0, 1 && \forall 0 \leq f \leq F-1, && \text{(3f)} \\
& && 0 \leq s \leq S-1, && \\
& && 0 \leq i \leq I-1 &&
\end{aligned}
$$

The objective function (3a) determines the utility of a given prioritisation: a combination of the total cost of a prioritisation and how fragmented it is. Constraints (3b) ensure that all the amount-based targets are met. Constraints (3c) ensure that all the space-based targets are met for each feature and each attribute space. For each feature and attribute space, the total weighted distance between the demand points and their closest selected planning units is calculated ($\lambda_{fsi} d_{fsij} Y_{fsij}$). This total weighted distance is then scaled by the inherent variation in the demand points ($\lambda_{fsi} \delta_{fsi}$). The resulting fraction yields a proportion conceptually similar to the statistical R^2 value, and the constraints ensure that this proportion must be greater than or equal to the space-based target. Constraints (3d) ensure that only one planning unit is assigned to each demand point. Constraints (3e) ensure that demand points are only assigned to selected planning units. Constraints (3f) ensure that the X and Y variables are binary.

Reliable formulation

The reliable formulation explicitly considers the probability that the planning units are inhabited. As a consequence, it may deliver prioritisations that will sufficiently represent an attribute space even if the features do not inhabit several of the planning units when the prioritisation is implemented. This behaviour is achieved by siting back-up planning units near selected planning units with low occupancy probabilities in the attribute space(s). To ensure that prioritisations are robust against multiple planning units being uninhabited, the problem assigns planning units at multiple backup levels.

Backup levels levels are defined as r-levels (similar to failure levels in Snyder and Daskin 2005). The first backup r-level is used to calculate the level of representation when all of the selected planning units are occupied by all $f \in F$. For this scenario, the closest selected planning unit to each demand point i for attribute space s is assigned at r-level$= 0$. This scenario essentially represents Y_{fsij} in the unreliable formulation. The second backup r-level is used to assess the level of representation when the closest planning unit to each demand point i is unoccupied. For this scenario, the second closest planning units are assigned at r-level$= 1$. The third backup r-level is used to assess representation when the first two closest planning units are unoccupied. The third closest planning units are assigned at r-level$= 2$. Continuing on, in this manner, the selected planning units in a prioritisation are assigned to each demand point $i \in I$, attribute space $s \in S$, and each feature $f \in F$ at an r-level.

A final backup r-level when $r = R$ is used to assess the level of representation when the features $f \in F$ do not occupy any selected planning units in a prioritisation. Each demand point $i \in I$ for each $s \in S$ and $f \in F$ is assigned to an "imaginary" planning unit $j = J$ at $r = R$. The distance variables associated with this imaginary planning unit d_{fsiJ} denote the loss of biological value associated with failing to secure a representative sample of feature f in attribute space s. However, the d variables are in distance units which are meaningless in this context. Thus these variables are calculated using a failure multiplier (M) and the maximum distance between the planning units and the demand points for $f \in F$, $s \in S$ (4).

$$d_{fsiJ} = M \max_{0 \leq i \leq I-1, 0 \leq j \leq J-1} d_{fsij} \qquad \forall 0 \leq f \leq F-1, \; 0 \leq s \leq S-1 \tag{4}$$

Moderately-sized conservation planning problems often include several thousand planning units. It is currently not be feasible to solve this problem when considering all possible failure scenarios. As a consequence, the R variable can be any $1 \leq R \leq J-1$. For instance, when $R = 3$ only 2 backup levels are considered in addition to the final backup level. Cui et al. (2010) found that $R = 5$ yields similar solutions to $R = J$ when $J >> 5$. However, depending on the number of features, demand points, attribute spaces, and planning units, decision makers will likely be limited to $R = 1$ to obtain prioritisations in a feasible amount of time.

In the reliable formulation, the control variables are the B (1a), T_s (1b), τ_{sa} (1c), R, and M variables.

$$R = \text{number of failure levels} \tag{5a}$$
$$M = \text{failure multiplier} \tag{5b}$$

The decision variables are the X_j (2a), Y_{fsijr}, P_{fsijr} variables.

$$Y_{fsijr} = \begin{cases} 1, & \begin{array}{l}\text{if demand point } i \text{ is assigned to planning unit } j \text{ for feature} \\ f \text{ in space } s \text{ at back-up level } r.\end{array} \\ 0, & \text{otherwise} \end{cases} \tag{6a}$$

$$P_{fsijr} = \begin{array}{l}\text{probability that demand point } i \text{ is assigned to planning} \\ \text{unit } j \text{ at back-up level } r \text{ for feature } f \text{ and space } s\end{array} \tag{6a}$$

The reliable formulation (RRAP) is a multi-objective optimisation problem.

$$
\begin{aligned}
\text{(RRAP)} \quad & \text{Min } (3a) && \\
& \text{s.t. } (3b) && \\
& 1 - \frac{\sum_{i=0}^{I-1} \sum_{j=0}^{J-1} \left(\lambda_{fsi} d_{fsij} P_{fsijr} Y_{fsij}\right)^2}{\sum_{i=0}^{I-1} \left(\lambda_{fsi} \delta_{fsi}\right)^2} \geq T_{fs} && \forall 0 \leq f \leq F-1, \quad (7a) \\
& && 0 \leq s \leq S-1 \\
& \sum_{j=0}^{J-1} Y_{fsijr} = 1 && \forall 0 \leq f \leq F-1, \quad (7b) \\
& && 0 \leq s \leq S-1, \\
& && 0 \leq i \leq I-1, \\
& && 0 \leq r \leq R \\
& \sum_{r=0}^{R} Y_{fsijr} = 1 && \forall 0 \leq f \leq F-1, \quad (7c) \\
& && 0 \leq s \leq S-1, \\
& && 0 \leq i \leq I-1, \\
& && 0 \leq j \leq J \\
& \sum_{r=0}^{R-1} Y_{fsijr} \leq X_j && \forall 0 \leq f \leq F-1, \quad (7d) \\
& && 0 \leq s \leq S-1, \\
& && 0 \leq i \leq I-1, \\
& && 0 \leq j \leq J-1 \\
& Y_{fsiJR} = 1 && \forall 0 \leq f \leq F-1, \quad (7e) \\
& && 0 \leq s \leq S-1, \\
& && 0 \leq i \leq I-1 \\
& P_{fsij0} = q_{fj} && \forall 0 \leq f \leq F-1, \quad (7f) \\
& && 0 \leq s \leq S-1, \\
& && 0 \leq i \leq I-1, \\
& && 0 \leq j \leq J \\
& P_{fsijr} = (1-) \sum_{k=0}^{J-1} \frac{1-q_k}{q_k} P_{f,s,i,k,r-1} Y_{f,s,i,k,r-1} && \forall 0 \leq f \leq F-1, \quad (7g) \\
& && 0 \leq s \leq S-1, \\
& && 0 \leq i \leq I-1, \\
& && 0 \leq j \leq J, \\
& && 1 \leq r \leq R \\
& X_j, Y_{fsijr} \in 0, 1 && \forall 0 \leq f \leq F-1, \quad (7h) \\
& && 0 \leq s \leq S-1, \\
& && 0 \leq i \leq I-1, \\
& && 0 \leq j \leq J, \\
& && 0 \leq r \leq R
\end{aligned}
$$

The objective function for the reliable formulation is the same as for the unreliable formation (3a). Similar to the unreliable formulation, constraints (3b) and (7a) ensure that the amount-based and space-based targets are met. Constraint (7b–7c) ensure that each planning unit is only assigned to one backup r-level for $i \in I$. Constraints (7d) ensure that only selected planning units are assigned to demand points $i \in I$. Constraints (7e) ensure that the imaginary planning unit is always assigned to the highest backup r-level. Constraints (7f–7g) determine the probability that planning unit j will be used to sample demand point $i \in I$ for $s \in S$ and $f \in F$ (see Cui *et al.* 2010 for more information). Constraints (7h) ensure that the X and Y variables are binary.

Optimisation

The unreliable and reliable formulations are non-linear. However, the non-linear components can be linearised using existing techniques. First, the expression $X_j X_k$ in (3a) can be linearised using methods described by Beyer et al. (2016). Second, the expression $P_{fsijr} Y_{jsijr}$ in (7a) can be linearised using techniques described by Sherali and Alameddine (1992) as implemented in Cui et al. (2010). Linearised versions of the problems can be solved using commercial exact algorithm solvers.

The **rapr** R package provides functions to express conservation planning data as an optimisation problems using linearised versions of the unreliable and reliable formulations. These optimisation problems can then be solved to generate prioritisations using the commercial `Gurobi` software suite (http://www.gurobi.com). Note that academics can obtain a license at no cost from the Gurobi website. After installing the `Gurobi` software suite, users will need to install the `Gurobi R` package. This `R` package can be installed on Windows, Mac OSX, and Linux operating systems.

Package overview

To load the **rapr** R package and learn more about the package, type the following code into R.

```
# load rapr R package
library(rapr)

# show package overview
?rapr
```

The **rapr** R package uses a range of S4 classes to store conservation planning data, parameters, and prioritisations (Table 1).

Table 1: Main classes in the **rapr** R package

Class name	Description
`ManualOpts`	place-holder class for manually specified solutions
`GurobiOpts`	stores parameters for solving optimisation problems using Gurobi
`RapUnreliableOpts`	stores control variables parameters for the unreliable problem formulation
`RapReliableOpts`	stores control variables for the reliable problem formulation

Class name	Description
`SimplePoints`	represents coordinates in an n-dimensional space
`DemandPoints`	stores the coordinates and weights for a given species and attribute space
`AttributeSpace`	stores the coordinates for planning units and the demand points for each species
`RapData`	stores the all the planning unit, species, and attribute space data
`RapUnsolved`	stores all the data, control variables, and parameters needed to generate prioritisations
`RapResults`	stores the prioritisations and summary statistics generated after solving a problem
`RapSolved`	stores the input data and output results

Package tutorial

This tutorial is designed to provide users with an understanding of how to use the `rapr R` package to generate and compare solutions. This tutorial uses several additional packages, so first we will run the following code to load them.

```
# load packages for tutorial
library(plyr)
library(dplyr)
library(ggplot2)
library(RandomFields)
library(rgeos)

# set seed for reproducibility
set.seed(500)
```

Simple simulated species

Data

To investigate the behaviour of the problem, we will generate prioritisations for three simulated species. We will use the unreliable formulation of the problem to understand the basics, and later move onto the reliable formulation. The first species (termed 'uniform') will represent a hyper-generalist. This species will inhabit all areas with equal probability. The second species (termed 'normal') will represent a species with a single range core. The third species (termed 'bimodal') will represent a species with two distinct ecotypes, each with their own range core. To reduce computational time for this example, we will use a 10×10 grid of square planning units.

```r
# make planning units
sim_pus <- sim.pus(100L)

# simulate species distributions
sim_spp <- lapply(
    c('uniform', 'normal', 'bimodal'),
    sim.species,
  n=1,
  x=sim_pus,
  res=1
)
```

Let's see what these species' distributions look like.

```r
# plot species
plot(
    stack(sim_spp),
    main=c('Uniform species','Normal species','Bimodal species'),
    addfun=function(){lines(sim_pus)},
    nc=3
)
```

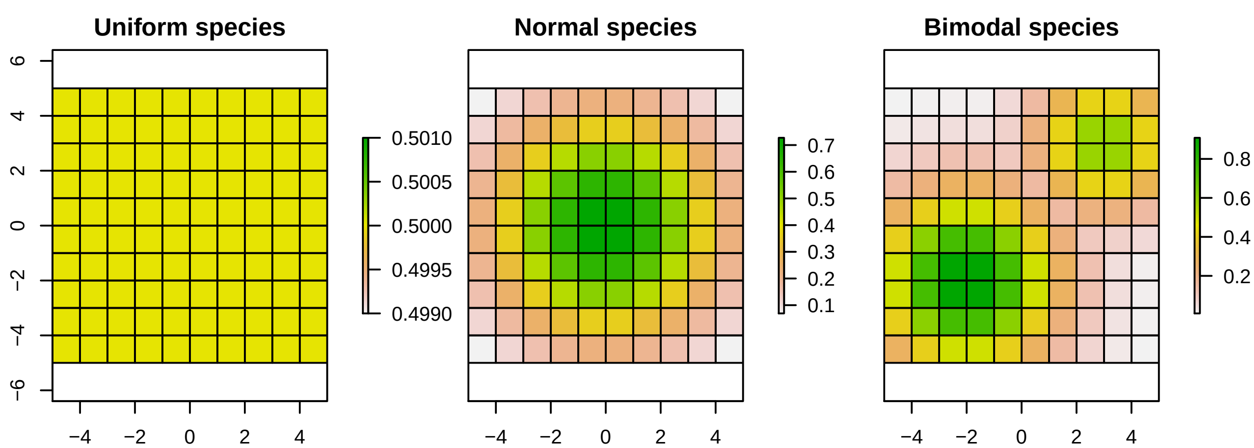


Figure 2 Distribution of three simulated species. Each square represents a planning unit. The colour of each square denotes the probability that individuals from each species occupy it.

Next, we will generate a set of demand points. To understand the effects of probabilities and weights on the demand points, we will generate the demand points in geographic space. These demand points will be the centroids of the planning units. Additionally, we will use the same set of demand points for each species and only vary the weights of the demand points between species. **Note that we are only using the same distribution of demand points for different species for teaching purposes. It is strongly recommended to use different demand points for different species in real-world conservation planning exercises.** See the case-study section of this tutorial for examples on how to generate suitable demand points.

```r
# generate coordinates for pus/demand points
pu_coords <- gCentroid(sim_pus, byid=TRUE)

# calculate weights
sim_dps <- lapply(
    sim_spp,
    function(x) {
        return(extract(x, pu_coords))
    }
)

# create demand point objects
sim_dps <- lapply(
    sim_dps,
    function(x) {
        return(
            DemandPoints(
                SimplePoints(pu_coords@coords),
                c(x)
            )
        )
    }
)
```

Now, we will construct a `RapUnsolved` object to store our input data and parameters. This contains all the information to generate prioritisations.

```r
## create RapUnreliableOpts object
# this stores parameters for the unreliable formulation problem (ie. BLM)
sim_ro <- RapUnreliableOpts()

## create RapData object
# create data.frame with species info
species <- data.frame(
  name=c('uniform', 'normal', 'bimodal')
)

## create data.frame with species and space targets
# amount targets at 20% (denoted with target=0)
# space targets at 20% (denoted with target=1)
targets <- expand.grid(
  species=1:3,
  target=0:1,
  proportion=0.2
)

# calculate probability of each species in each pu
```

```r
pu_probabilities <- calcSpeciesAverageInPus(sim_pus, stack(sim_spp))

## create AttributeSpace object
# this stores the coordinates of the planning units in an attribute space
# and the coordinates and weights of demand points in the space
attr_space <- AttributeSpace(
  SimplePoints(pu_coords@coords),
  sim_dps
)

# generate boundary data information
boundary <- calcBoundaryData(sim_pus)

## create RapData object
# this stores all the input data for the prioritisation
sim_rd <- RapData(
  sim_pus@data,
  species,
  targets,
  pu_probabilities,
  list(attr_space),
  boundary,
  SpatialPolygons2PolySet(sim_pus)
)

## create RapUnsolved object
# this stores all the input data and parameters needed to generate prioritisations
sim_ru <- RapUnsolved(sim_ro, sim_rd)
```

Single-species prioritisations

Amount-based targets

To investigate the effects of space-based targets, we will generate a prioritisation for each species using only amount-based targets and compare them to prioritisations generated using amount- and space-based targets. To start off, we will generate a prioritisation for the uniform species using amount-based targets. To do this, we will generate a new `sim_ru` object by subsetting out the data for the uniform species from the `sim_ru` object. Then, we will update the targets in the new object. Finally, we will solve the object to generate a prioritisation that fulfills the targets for minimal cost.

```r
# create new object with just the uniform species
sim_ru_s1 <- spp.subset(sim_ru, 'uniform')

# update amount targets to 20% and space targets to 0%
sim_ru_s1 <- update(sim_ru_s1, amount.target=0.2, space.target=NA, solve=FALSE)
```

```r
# solve problem to identify prioritisation
sim_rs_s1_amount <- solve(sim_ru_s1)
```

```
## Optimize a model with 1 rows, 100 columns and 100 nonzeros
## Coefficient statistics:
##   Matrix range    [5e-01, 5e-01]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+01, 1e+01]
## Found heuristic solution: objective 20
## Presolve removed 1 rows and 100 columns
## Presolve time: 0.00s
## Presolve: All rows and columns removed
##
## Explored 0 nodes (0 simplex iterations) in 0.00 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 2.000000000000e+01, best bound 2.000000000000e+01, gap 0.0%
```

```
## Warning in validityMethod(object): object@space.held contains values less
## than 0, some species are really poorly represented
```

```r
## show summary
# note the format for this is similar to that used by Marxan
# see ?rapr::summary for details on this table
summary(sim_rs_s1_amount)
```

```
##   Run_Number Status Score Cost Planning_Units Connectivity_Total
## 1          1 MANUAL    20   20             20                220
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1              42               168               10
##   Connectivity_In_Fraction
## 1                0.1909091
```

```r
# show amount held
amount.held(sim_rs_s1_amount)
```

```
##   uniform
## 1     0.2
```

```r
# show space held
space.held(sim_rs_s1_amount)
```

```
##   uniform (Space 1)
## 1        -0.2363636
```

Now that we have generated a prioritisation, let's see what it looks like. We can use the spp.plot method to see how the prioritisation overlaps with the uniform species' distribution. Note that since all planning units have equal probabilities for this species, all planning units have the same fill colour.

```
# plot the prioritisation and the uniform species' distribution
spp.plot(sim_rs_s1_amount, 1, main='Uniform species')
```

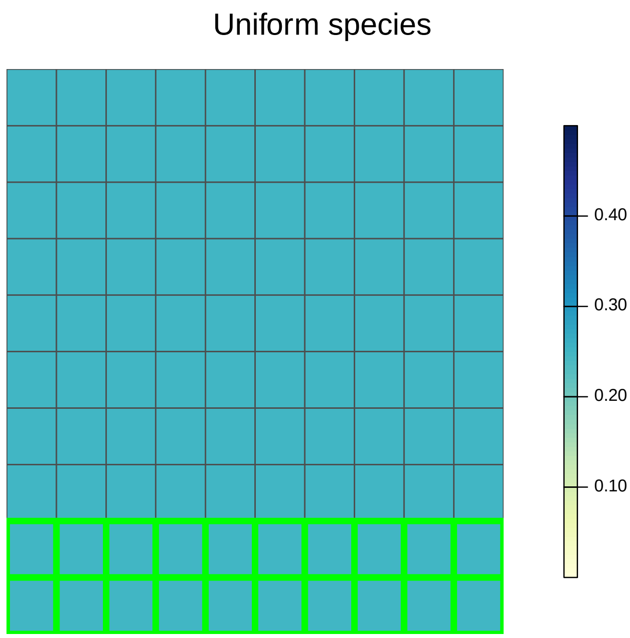


Figure 3 A prioritisation for the uniformly distributed species generated using amount-based targets (20%). Sqaures represent planning units. Planning units with a green border are selected for prioritisation, and their colour denotes the probability they are inhabited by the species.

The prioritisation for the uniform species appears to be just a random selection of planning units. This behavior is due to the fact that any prioritisation with 20 planning units is optimal. By relying on just amount targets, this solution may preserve a section of the species' range core, or just focus on the range margin, or some random part of its range–no emphasis is directed towards preserving different parts of the species' range. This behavior highlights a fundamental limitation of just using amount-based targets. In the absence of additional criteria, conventional reserve selection problems do not contain any additional information to identify the most effective prioritisation.

Now, we will generate a prioritisation for the normally distributed species using amount-based targets. We will use a similar process to what we used for the uniformly distributed species, but for brevity, we will use code to generate solutions immediately after updating the object.

```
# create new object with just the normal species
sim_ru_s2 <- spp.subset(sim_ru, 'normal')
```

```r
# update amount targets to 20% and space targets to 0% and solve it
sim_rs_s2_amount <- update(sim_ru_s2, amount.target=0.2, space.target=NA, solve=TRUE)
```

```
## Optimize a model with 1 rows, 100 columns and 100 nonzeros
## Coefficient statistics:
##   Matrix range    [7e-02, 7e-01]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [7e+00, 7e+00]
## Found heuristic solution: objective 27
## Presolve removed 0 rows and 86 columns
## Presolve time: 0.01s
## Presolved: 1 rows, 14 columns, 14 nonzeros
## Variable types: 0 continuous, 14 integer (0 binary)
## Presolved: 1 rows, 14 columns, 14 nonzeros
##
##
## Root relaxation: objective 9.864476e+00, 6 iterations, 0.01 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0    9.86448    0    1   27.00000    9.86448  63.5%     -    0s
## H    0     0                      10.0000000    9.86448  1.36%     -    0s
##
## Explored 0 nodes (6 simplex iterations) in 0.03 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 1.000000000000e+01, best bound 1.000000000000e+01, gap 0.0%
```

```r
# show summary
summary(sim_rs_s2_amount)
```

```
##   Run_Number Status Score Cost Planning_Units Connectivity_Total
## 1          1 MANUAL    10   10             10                220
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1              12               192               16
##   Connectivity_In_Fraction
## 1               0.05454545
```

```r
# show amount held
amount.held(sim_rs_s2_amount)
```

```
##      normal
## 1 0.2026153
```

```r
# show space held
space.held(sim_rs_s2_amount)
```

```
##   normal (Space 1)
## 1        0.6519926
```

Now let's visualise the prioritisation we made for the normal species.

```r
# plot the prioritisation and the normal species' distribution
spp.plot(sim_rs_s2_amount, 1, main='Normal species')
```

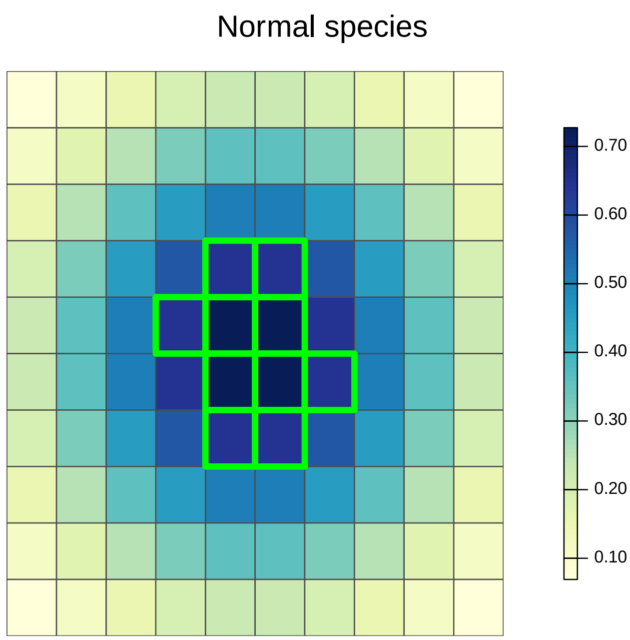


Figure 4 A prioritisation for the normally distributed species generated using amount-based targets (20%). See Figure 3 caption for conventions.

The amount-based prioritisation for the normal species focuses only on the species' range core. This prioritisation fails to secure any peripheral parts of the species' distribution. As a consequence, it may miss out on populations with novel adaptations to environmental conditions along the species' range margin.

Now, let's generate an amount-based target for the bimodally distributed species view it.

```r
# create new object with just the bimodal species
sim_ru_s3 <- spp.subset(sim_ru, 'bimodal')
```

```r
# update amount targets to 20% and space targets to 0% and solve it
sim_rs_s3_amount <- update(sim_ru_s3, amount.target=0.2, space.target=NA)
```

```
## Optimize a model with 1 rows, 100 columns and 100 nonzeros
## Coefficient statistics:
##   Matrix range    [7e-03, 9e-01]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [7e+00, 7e+00]
## Found heuristic solution: objective 21
## Presolve removed 0 rows and 75 columns
## Presolve time: 0.00s
## Presolved: 1 rows, 25 columns, 25 nonzeros
## Variable types: 0 continuous, 25 integer (0 binary)
## Presolved: 1 rows, 25 columns, 25 nonzeros
##
##
## Root relaxation: objective 7.919039e+00, 18 iterations, 0.00 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0    7.91904    0    1   21.00000    7.91904  62.3%     -    0s
## H    0     0                       8.0000000    7.91904  1.01%     -    0s
##
## Explored 0 nodes (18 simplex iterations) in 0.00 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 8.000000000000e+00, best bound 8.000000000000e+00, gap 0.0%
```

```
# plot the prioritisation and the bimodal species' distribution
spp.plot(sim_rs_s3_amount, 1, main='Bimodal species')
```

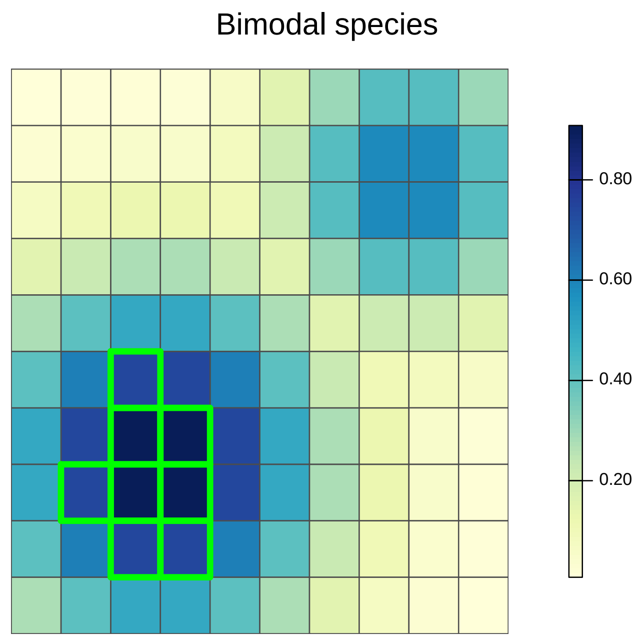


Figure 5 A prioritisation for the bimodally distributed species generated using amount-based targets (20%). See Figure 3 caption for conventions.

```r
# show summary
summary(sim_rs_s3_amount)
```

```
##   Run_Number Status Score Cost Planning_Units Connectivity_Total
## 1          1 MANUAL     8    8              8                220
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1               9               197               14
##   Connectivity_In_Fraction
## 1               0.04090909
```

```r
# show amount held
amount.held(sim_rs_s3_amount)
```

```
##     bimodal
## 1 0.2018391
```

```r
# show space held
space.held(sim_rs_s3_amount)
```

```
##   bimodal (Space 1)
## 1         0.2829105
```

The amount-based prioritisation for the bimodally distributed species only selects planning units in the bottom left corner of the study area. This prioritisation only preserves individuals belonging to one of the two ecotypes. As a consequence, this prioritisation may fail to preserve a representative sample of the genetic variation found inside this species.

Amount-based and space-based targets

Now that we have generated a prioritisation for each species using only amount-based targets, we will generate a prioritisations using both amount-based and space-targets. To do this we will update the space targets in our amount-based prioritisations to 85%, and store the new prioritisations in new objects.

First, let's do this for the uniform species.

```
# make new prioritisation
sim_rs_s1_space <- update(sim_rs_s1_amount, amount.target=0.2, space.target=0.85)
```

```
## Warning for adding variables: zero or small (< 1e-13) coefficients, ignored
## Optimize a model with 10102 rows, 10100 columns and 40000 nonzeros
## Coefficient statistics:
##   Matrix range    [2e-01, 4e+01]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 6e+01]
## Found heuristic solution: objective 89
## Presolve removed 36 rows and 0 columns
## Presolve time: 4.24s
## Presolved: 10066 rows, 10100 columns, 40104 nonzeros
## Variable types: 0 continuous, 10100 integer (10100 binary)
## Presolved: 10066 rows, 10100 columns, 40104 nonzeros
##
## Presolve removed 10066 rows and 10100 columns
##
## Root relaxation: objective 2.000000e+01, 621 iterations, 0.26 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
## *    0     0               0      20.0000000   20.00000  0.00%     -    4s
##
## Explored 0 nodes (1086 simplex iterations) in 4.66 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 2.000000000000e+01, best bound 2.000000000000e+01, gap 0.0%
```

```
# show summary
summary(sim_rs_s1_space)
```

```
##   Run_Number Status Score Cost Planning_Units Connectivity_Total
```

```
## 1          1 MANUAL    20   20             20                220
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1              18               147               55
##   Connectivity_In_Fraction
## 1               0.08181818
```

```r
# show amount held
amount.held(sim_rs_s1_space)
```

```
##   uniform
## 1     0.2
```

```r
# show space held
space.held(sim_rs_s1_space)
```

```
##   uniform (Space 1)
## 1         0.8793939
```

Let's take a look at the prioritisation for the uniform species with amount-based and space-based targets. Then, let's compare the solutions for the amount-based prioritisation with the new prioritisation using both amount and space targets.

```r
# plot the prioritisation and the uniform species' distribution
spp.plot(sim_rs_s1_space, 'uniform', main='Uniform species')
```

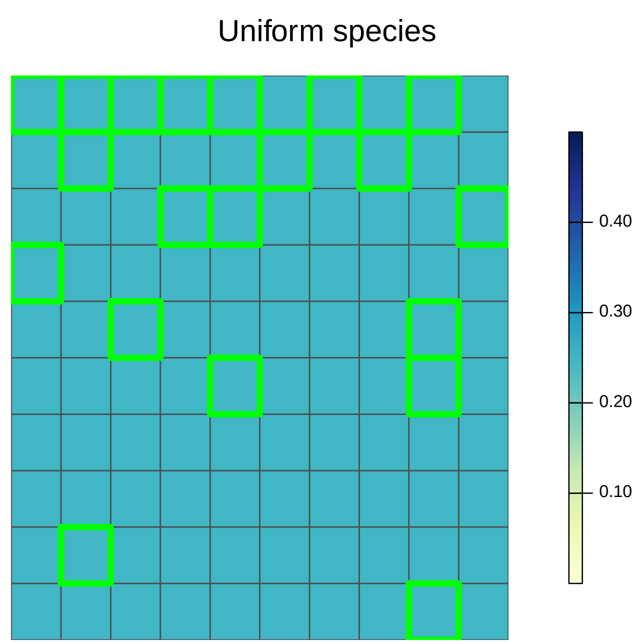


Figure 6 A prioritisation for the uniformly distributed species generated using amount-based targets (20%) and space-based targets (85%). See Figure 3 caption for conventions.

```r
# plot the difference between old and new prioritisations
plot(sim_rs_s1_amount, sim_rs_s1_space, 1, 1, main='Difference between solutions')
```

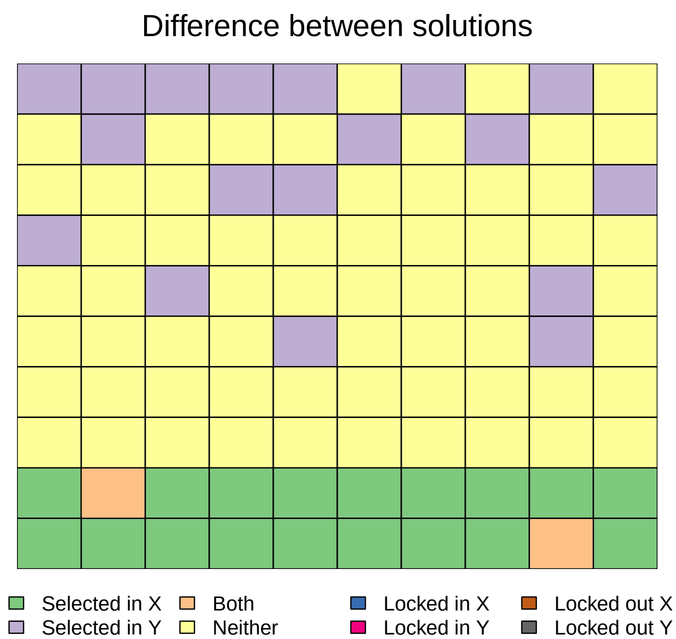


Figure 7 Difference between two prioritisations for the uniformly distributed species. Prioritisation X was generated using just amount-based targets (20%), and prioritisation Y was generated using an additional space-based target (85%).

Here we can see that by including a space-target, the prioritisation is spread out evenly across the species' distribution. Unlike the amount-based prioritisation, this prioritisation samples all the different parts of the species' distribution.

Now, let's generate a prioritisation for the normally distributed species that considers amount-based and space-based targets. Then, let's visualise the new prioritisation and compare it to the old amount-based prioritisation.

```r
# make new prioritisation
sim_rs_s2_space <- update(sim_rs_s2_amount, amount.target=0.2, space.target=0.85)
```

```
## Warning for adding variables: zero or small (< 1e-13) coefficients, ignored
## Optimize a model with 10102 rows, 10100 columns and 40000 nonzeros
## Coefficient statistics:
##   Matrix range    [5e-03, 3e+01]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 2e+01]
## Found heuristic solution: objective 87
## Presolve removed 1176 rows and 192 columns (presolve time = 5s) ...
```

```
## Presolve removed 1178 rows and 192 columns (presolve time = 10s) ...
## Presolve removed 1178 rows and 192 columns
## Presolve time: 10.93s
## Presolved: 8924 rows, 9908 columns, 55525 nonzeros
## Variable types: 0 continuous, 9908 integer (9908 binary)
## Presolve removed 1 rows and 0 columns
## Presolved: 8923 rows, 9908 columns, 55510 nonzeros
##
## Presolve removed 8923 rows and 9908 columns
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    1.0000000e+02   0.000000e+00   1.000000e+02     11s
##     3156    1.2049877e+01   0.000000e+00   0.000000e+00     12s
##     3156    1.2049877e+01   0.000000e+00   0.000000e+00     12s
##
## Root relaxation: objective 1.204988e+01, 3156 iterations, 0.90 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0   12.04988    0  409   87.00000   12.04988  86.1%     -   12s
## H    0     0                      13.0000000   12.04988  7.31%     -   13s
##
## Explored 0 nodes (3431 simplex iterations) in 13.06 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 1.300000000000e+01, best bound 1.300000000000e+01, gap 0.0%
```

```r
# show summary
summary(sim_rs_s2_space)
```

```
##   Run_Number Status Score Cost Planning_Units Connectivity_Total
## 1          1 MANUAL    13   13             13                220
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1               7               175               38
##   Connectivity_In_Fraction
## 1               0.03181818
```

```r
# show amount held
amount.held(sim_rs_s2_space)
```

```
##      normal
## 1 0.2122983
```

```r
# show space held
space.held(sim_rs_s2_space)
```

```
##   normal (Space 1)
## 1        0.8523318
```

```r
# plot the prioritisation and the normal species' distribution
spp.plot(sim_rs_s2_space, 'normal', main='Normal species')
```

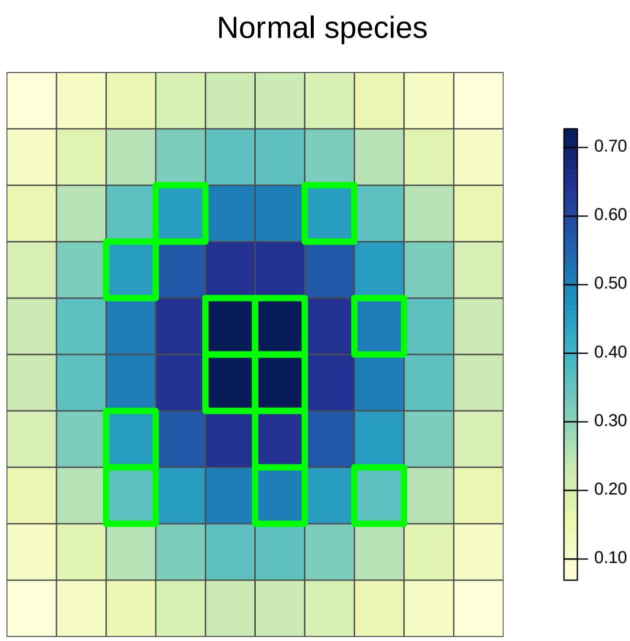


Figure 8 A prioritisation for the normally distributed species generated using amount-based targets (20%) and space-based targets (85%). See Figure 3 caption for conventions.

```r
# plot the difference between old and new prioritisations
plot(sim_rs_s2_amount, sim_rs_s2_space, 1, 1, main='Difference between solutions')
```

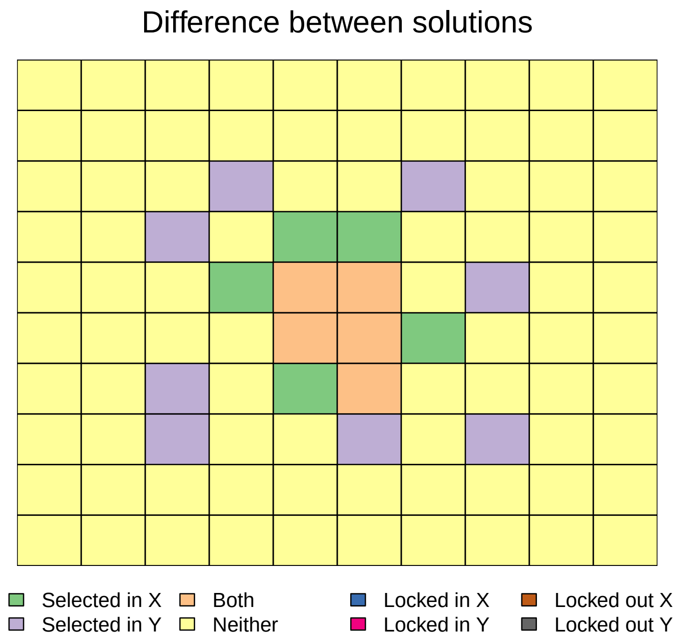


Figure 9 Difference between two prioritisations for the normally distributed species. See Figure 7 caption for conventions.

We can see by using both amount-based and space-based targets we can obtain a prioritisation that secures both the species' range core and parts of its range margin. As a consequence, it may capture any novel adaptations found along the species' range margin–unlike the amount-based prioritisation.

Finally, let's generate a prioritisation for the bimodal species using amount-based and space-based targets.

```
# make new prioritisation
sim_rs_s3_space <- update(sim_rs_s3_amount, amount.target=0.2, space.target=0.85)
```

```
## Warning for adding variables: zero or small (< 1e-13) coefficients, ignored
## Optimize a model with 10102 rows, 10100 columns and 40000 nonzeros
## Coefficient statistics:
##   Matrix range    [6e-05, 8e+01]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 3e+01]
## Found heuristic solution: objective 83
## Presolve removed 817 rows and 184 columns (presolve time = 5s) ...
## Presolve removed 817 rows and 184 columns
## Presolve time: 7.79s
## Presolved: 9285 rows, 9916 columns, 50895 nonzeros
## Variable types: 0 continuous, 9916 integer (9916 binary)
## Presolve removed 2 rows and 0 columns
## Presolved: 9283 rows, 9916 columns, 50869 nonzeros
```

```
##
## Presolve removed 9283 rows and 9916 columns
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    1.0000000e+02   0.000000e+00   1.000000e+02      8s
##     9465    8.1205830e+00   0.000000e+00   0.000000e+00     10s
##     9465    8.1205830e+00   0.000000e+00   0.000000e+00     10s
##
## Root relaxation: objective 8.120583e+00, 9465 iterations, 2.09 seconds
## Total elapsed time = 10.02s
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0    8.12058    0  207   83.00000    8.12058  90.2%     -   10s
## H    0     0                       9.0000000    8.12058  9.77%     -   10s
##
## Explored 0 nodes (9665 simplex iterations) in 10.32 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 9.000000000000e+00, best bound 9.000000000000e+00, gap 0.0%
```

```r
# show summary
summary(sim_rs_s3_space)
```

```
##   Run_Number Status Score Cost Planning_Units Connectivity_Total
## 1          1 MANUAL     9    9              9                220
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1               9               193               18
##   Connectivity_In_Fraction
## 1               0.04090909
```

```r
# show amount held
amount.held(sim_rs_s3_space)
```

```
##    bimodal
## 1 0.219729
```

```r
# show space held
space.held(sim_rs_s3_space)
```

```
##   bimodal (Space 1)
## 1         0.8813757
```

```r
# plot the prioritisation and the bimodal species' distribution
spp.plot(sim_rs_s3_space, 'bimodal', main='Bimodal species')
```

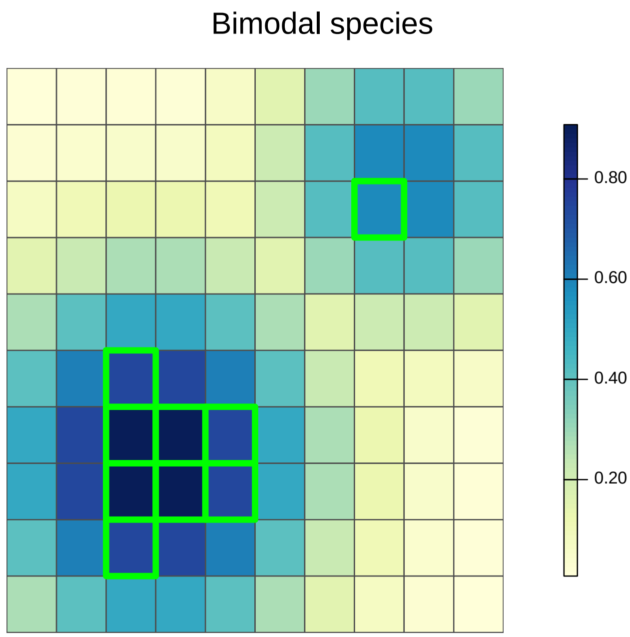


Figure 10 A prioritisation for the normally distributed species generated using amount-based targets (20%) and space-based targets (85%). See Figure 3 caption for conventions.

```r
# plot the difference between old and new prioritisations
plot(sim_rs_s3_amount, sim_rs_s3_space, 1, 1, main='Difference between solutions')
```

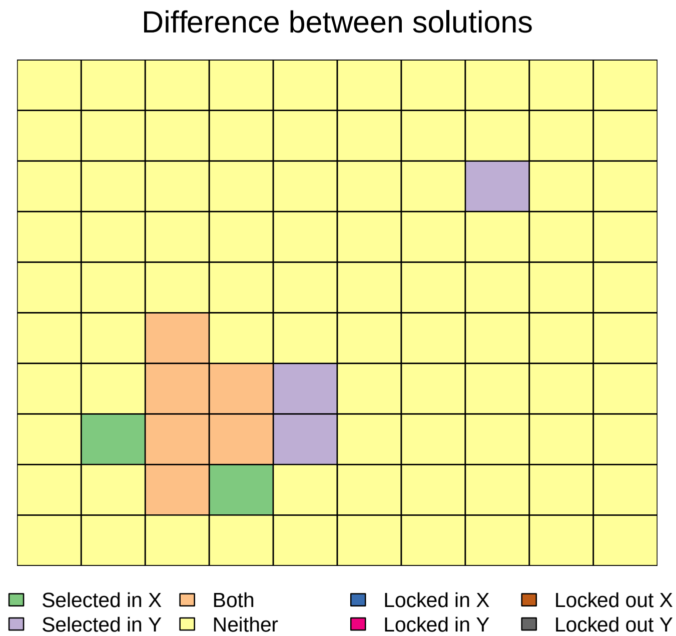


Figure 11 Difference between two prioritisations for the bimodally distributed species. See Figure 7 caption for conventions.

Earlier we found that the amount-based prioritisation only preserved individuals from a single ecotype, and would have failed to adequately preserve the intra-specific variation for this species. However, here we can see that by including space-based targets, we can develop prioritisations that secure individuals belonging to both ecotypes. This new prioritisation is much more effective at sampling the intra-specific variation for this species.

Overall, these results demonstrate that under the simplest of conditions, the use of space-based targets can improve prioritisations. However, these prioritisations were each generated for a single species. Prioritisations generated using multiple species may do a better job at preserving the intra-specific variation for individuals species by preserving them in different parts of their range. We will investigate this in the next section.

Multi-species prioritisations

Effects of including space-based targets

So far we have generated prioritisations using only a single species at a time. However, real world prioritisations are often generated using multiple species to ensure that they preserve a comprehensive set of biodiversity. Here, we will generate multi-species prioritisations that preserve all three of the simulated species. First, we will generate a prioritisation using amount-based targets that only aims to preserve 20% of the area they occupy. Then, we will generate a prioritisation that also incoperate space-based targets to also preserve 85% of their geographic distribution. We will then compare the two prioritisations.

```
# make prioritisations
sim_mrs_amount <- update(
    sim_ru,
```

```r
    amount.target=c(0.2,0.2,0.2),
    space.target=c(0,0,0)
)
```

```
## Warning for adding variables: zero or small (< 1e-13) coefficients, ignored
## Optimize a model with 30306 rows, 30100 columns and 120000 nonzeros
## Coefficient statistics:
##   Matrix range     [6e-05, 8e+01]
##   Objective range  [1e+00, 1e+00]
##   Bounds range     [1e+00, 1e+00]
##   RHS range        [1e+00, 4e+02]
## Found heuristic solution: objective 99
## Presolve removed 0 rows and 0 columns (presolve time = 5s) ...
## Presolve time: 5.44s
## Presolved: 30306 rows, 30100 columns, 120000 nonzeros
## Variable types: 0 continuous, 30100 integer (30100 binary)
## Presolved: 30306 rows, 30100 columns, 120000 nonzeros
##
## Presolve removed 30306 rows and 30100 columns
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    1.0000000e+02   0.000000e+00   1.000000e+02      7s
##     2176    2.0000000e+01   3.848761e+02   0.000000e+00     10s
##     3319    2.0000000e+01   0.000000e+00   0.000000e+00     12s
##     3319    2.0000000e+01   0.000000e+00   0.000000e+00     12s
##
## Root relaxation: objective 2.000000e+01, 3319 iterations, 6.65 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
## *    0     0               0      20.0000000   20.00000  0.00%     -   12s
##
## Explored 0 nodes (4618 simplex iterations) in 12.35 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 2.000000000000e+01, best bound 2.000000000000e+01, gap 0.0%
```

```r
sim_mrs_space <- update(
    sim_ru,
    amount.target=c(0.2,0.2,0.2),
    space.target=c(0.85, 0.85, 0.85)
)
```

```
## Warning for adding variables: zero or small (< 1e-13) coefficients, ignored
## Optimize a model with 30306 rows, 30100 columns and 120000 nonzeros
## Coefficient statistics:
##   Matrix range     [6e-05, 8e+01]
##   Objective range  [1e+00, 1e+00]
##   Bounds range     [1e+00, 1e+00]
##   RHS range        [1e+00, 6e+01]
## Found heuristic solution: objective 99
## Presolve removed 1362 rows and 376 columns (presolve time = 5s) ...
## Presolve removed 1391 rows and 376 columns (presolve time = 10s) ...
## Presolve removed 1593 rows and 376 columns (presolve time = 15s) ...
## Presolve removed 1636 rows and 376 columns (presolve time = 20s) ...
## Presolve removed 1636 rows and 376 columns (presolve time = 25s) ...
## Presolve removed 1636 rows and 376 columns
## Presolve time: 28.43s
## Presolved: 28670 rows, 29724 columns, 137500 nonzeros
## Variable types: 0 continuous, 29724 integer (29724 binary)
## Presolved: 28670 rows, 29724 columns, 137500 nonzeros
##
## Presolve removed 28670 rows and 29724 columns
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    1.0000000e+02   0.000000e+00   1.000000e+02     30s
##      864    5.3831797e+01   0.000000e+00   3.188844e+03     30s
##     4125    2.0000000e+01   0.000000e+00   0.000000e+00     32s
##     4125    2.0000000e+01   0.000000e+00   0.000000e+00     32s
##
## Root relaxation: objective 2.000000e+01, 4125 iterations, 3.76 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0   20.00000    0    2   99.00000   20.00000  79.8%     -    32s
## H    0     0                      20.0000000   20.00000  0.00%     -    32s
##
## Explored 0 nodes (5947 simplex iterations) in 32.99 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 2.000000000000e+01, best bound 2.000000000000e+01, gap 0.0%
```

```
# show summaries
summary(sim_mrs_amount)
```

```
##   Run_Number Status Score Cost Planning_Units Connectivity_Total
```

```
## 1          1 MANUAL    20   20             20                 220
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1              17               147               56
##   Connectivity_In_Fraction
## 1               0.07727273
```

```r
summary(sim_mrs_space)
```

```
##   Run_Number Status Score Cost Planning_Units Connectivity_Total
## 1          1 MANUAL    20   20             20                 220
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1               7               142               71
##   Connectivity_In_Fraction
## 1               0.03181818
```

```r
# show amount held for each prioritisation
amount.held(sim_mrs_amount)
```

```
##   uniform   normal   bimodal
## 1     0.2 0.201559 0.2012952
```

```r
amount.held(sim_mrs_space)
```

```
##   uniform    normal   bimodal
## 1     0.2 0.2185579 0.2232897
```

```r
# show space held for each prioritisation
space.held(sim_mrs_amount)
```

```
##   uniform (Space 1) normal (Space 1) bimodal (Space 1)
## 1         0.8593939        0.8205165         0.8866593
```

```r
space.held(sim_mrs_space)
```

```
##   uniform (Space 1) normal (Space 1) bimodal (Space 1)
## 1         0.9321212        0.8805152         0.9261063
```

```r
# plot multi-species prioritisation with amount-based targets
plot(sim_mrs_amount, 1, main='Amount-based targets')
```

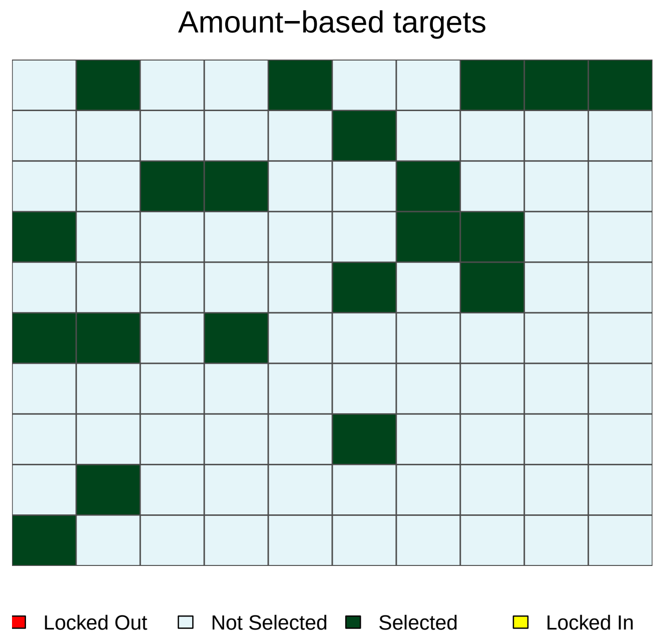


Figure 12 A multi-species prioritisation for the uniformly, normally, and bimodally distributed species generated using just amount-based targets (20%). Squares represent planning units. Dark green planning units are selected for preservation.

```
# plot multi-species prioritisation with amount- and space-based targets
plot(sim_mrs_space, 1, main='Amount and space-based targets')
```

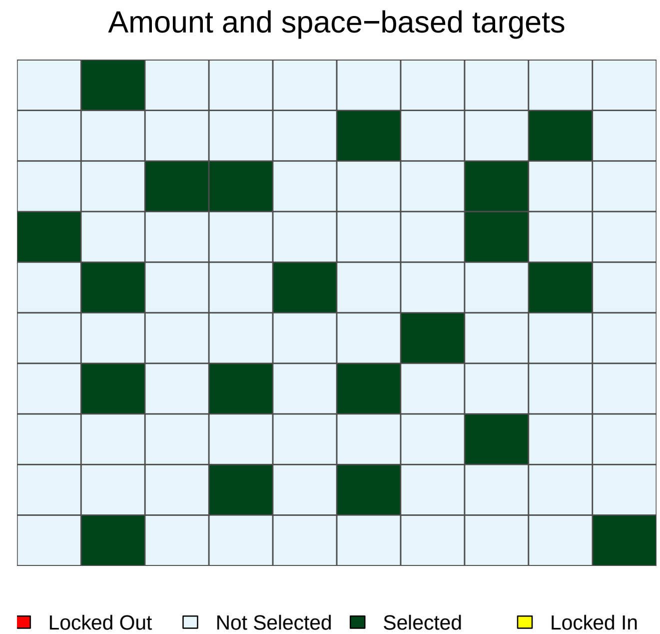


Figure 13 A multi-species prioritisation for the uniformly, normally, and bimodally distributed species generated using amount-based targets (20%) and space-based targets (85%). See Figure 12 caption for conventions.

```
# difference between the two prioritisations
plot(sim_mrs_amount, sim_mrs_space, 1, 1, main='Difference between solutions')
```

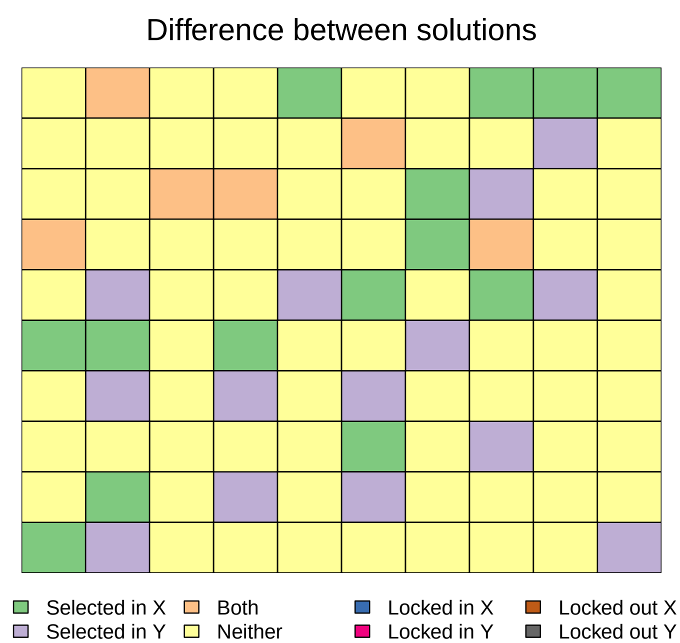


Figure 14 Difference between two multi-species prioritisations. See Figure 7 caption for conventions.

Here we can see that the inclusion of space-based targets changes which planning units are selected for prioritisation, but also the number of planning units that are selected. The amount-based prioritisation is comprised of 20 units, and the space-based prioritisation is comprised of 20 units. This result suggests that an adequate and representative prioritisation can be achieved for only a minor increase in cost.

Uncertainty in species' distributions

The unreliable formulation does not consider the probability that the planning units are occupied by features when calculating how well a given solution secures a representative sample of an attribute space. Thus solutions identified using the unreliable formulation may select regions of an attribute space for a species using planning units that only have a small chance of being inhabited. As a consequence, if the prioritisation is implemented, it may fail to secure regions of an attribute space if individuals do not inhabit these planning units, and ultimately fail to fulfil the space-based targets.

A simple solution to this issue would be to ensure that planning units cannot be assigned to demand points if they have a low probability of occupancy. This can be achieved by setting a probability threshold for planning units, such that planning units with a probability of occupancy below the threshold are effectively set to zero.

```
# make new prioritisation with probability threshold of 0.5 for each species
sim_mrs_space2 <- solve(
    prob.subset(
        sim_mrs_space,
        species=1:3,
        threshold=c(0.1,0.1,0.1)
    )
)
```

```
## Warning for adding variables: zero or small (< 1e-13) coefficients, ignored
## Optimize a model with 27706 rows, 27500 columns and 109600 nonzeros
## Coefficient statistics:
##   Matrix range    [3e-04, 8e+01]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 6e+01]
## Found heuristic solution: objective 99
## Presolve removed 1197 rows and 270 columns (presolve time = 5s) ...
## Presolve removed 1219 rows and 270 columns (presolve time = 10s) ...
## Presolve removed 1326 rows and 270 columns (presolve time = 15s) ...
## Presolve removed 1366 rows and 270 columns (presolve time = 20s) ...
## Presolve removed 1366 rows and 270 columns (presolve time = 25s) ...
## Presolve removed 1366 rows and 270 columns
## Presolve time: 25.75s
## Presolved: 26340 rows, 27230 columns, 125602 nonzeros
## Variable types: 0 continuous, 27230 integer (27230 binary)
## Presolve removed 30 rows and 0 columns
## Presolved: 26310 rows, 27230 columns, 125523 nonzeros
##
```

```
## Presolve removed 26310 rows and 27230 columns
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    1.0000000e+02   0.000000e+00   1.000000e+02     28s
##     3575    2.2225740e+01   0.000000e+00   1.419211e+03     30s
##     4330    2.0000000e+01   0.000000e+00   0.000000e+00     31s
##     4330    2.0000000e+01   0.000000e+00   0.000000e+00     31s
##
## Root relaxation: objective 2.000000e+01, 4330 iterations, 5.13 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0   20.00000    0  364   99.00000   20.00000  79.8%     -   33s
## H    0     0                      22.0000000   20.00000  9.09%     -   33s
## H    0     0                      20.0000000   20.00000  0.00%     -   33s
##
## Explored 0 nodes (7172 simplex iterations) in 33.94 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 2.000000000000e+01, best bound 2.000000000000e+01, gap 0.0%
```

```r
# show summary
summary(sim_mrs_space2)
```

```
##   Run_Number Status Score Cost Planning_Units Connectivity_Total
## 1          1 MANUAL    20   20             20                220
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1               7               143               70
##   Connectivity_In_Fraction
## 1               0.03181818
```

```r
# plot prioritisation
plot(sim_mrs_space2, 1)
```

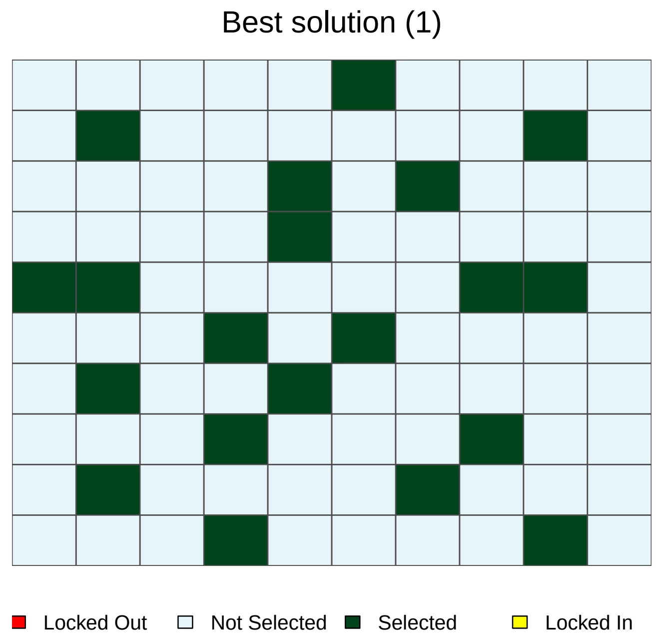


Figure 15 A multi-species prioritisation for the uniformly, normally, and bimodally distributed species generated using amount-based targets (20%) and space-based targets (85%). This priorititisation was generated to be robust against low occupancy probabilities, by preventing planning units with low probabilities from being used to represent demand points. See Figure 12 caption for conventions.

```
# difference between prioritisations that use and do not use thresholds
plot(sim_mrs_space2, sim_mrs_space, 1, 1, main='Difference between solutions')
```

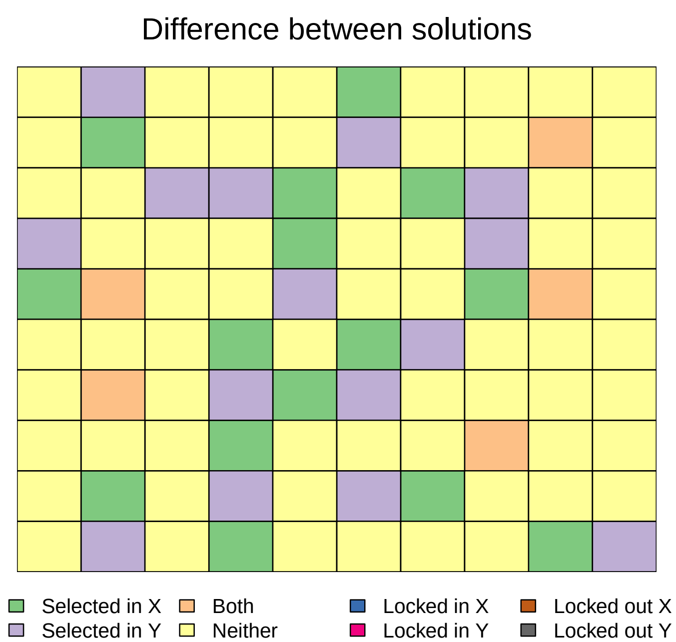


Figure 16 Difference between two multi-species prioritisations. See Figure 7 caption for conventions.

But this method requires setting somewhat arbitrary thresholds. A more robust solution to this issue is to actually use the probability that species occupy planning units to generate the prioritisations. This is what the reliable formulation does. First we will try and generate a solution using existing targets and the reliable formulation. To reduce computational time, we will set the maximum backup R-level to 1.

```
# make new prioritisation using reliable formulation
sim_mrs_space3 <- try(update(sim_mrs_space, formulation='reliable', max.r.level=1L))
```

```
## Warning for adding variables: zero or small (< 1e-13) coefficients, ignored
## Optimize a model with 364206 rows, 181900 columns and 3847200 nonzeros
## Coefficient statistics:
##   Matrix range    [6e-05, 1e+02]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [7e-03, 6e+01]
## Presolve removed 349813 rows and 90706 columns
## Presolve time: 3.98s
##
## Explored 0 nodes (0 simplex iterations) in 6.76 seconds
## Thread count was 1 (of 2 available processors)
##
## Model is infeasible
## Best objective -, best bound -, gap -
## Try setting lower space-based targets.
##  Below are the maximum targets for each species and space.
##   Proportion            Target
## 1   -1.97000 uniform (Space 1)
## 2  -12.67570  normal (Space 1)
## 3  -21.69092 bimodal (Space 1)
```

However, this fails. The reason why we cannot generate a prioritisation that fulfills these targets is because even the solution that contains all the planning units is still insufficient when we consider probabilities. The negative maximum targets printed in the error message indicate that planning units have low probabilities of occupancy. To fulfill the targets, we must obtain more planning units with higher probabilities of occupancy. We also could attempt resolving the problem using a higher R-level. Instead, we will set lower targets and generate solution.

```
# make new prioritisation using reliable formulation and reduced targets
sim_mrs_space3 <- update(
    sim_mrs_space,
    formulation='reliable',
    max.r.level=1L,
    space.target=-25
)
```

```
## Warning for adding variables: zero or small (< 1e-13) coefficients, ignored
```

```
## Optimize a model with 364206 rows, 181900 columns and 3847200 nonzeros
## Coefficient statistics:
##   Matrix range     [6e-05, 1e+02]
##   Objective range  [1e+00, 1e+00]
##   Bounds range     [1e+00, 1e+00]
##   RHS range        [7e-03, 1e+04]
## Presolve removed 333501 rows and 90800 columns (presolve time = 5s) ...
## Presolve removed 333701 rows and 151600 columns (presolve time = 10s) ...
## Presolve removed 333701 rows and 151600 columns (presolve time = 15s) ...
## Presolve removed 333701 rows and 151600 columns (presolve time = 113s) ...
## Presolve removed 333701 rows and 151600 columns (presolve time = 115s) ...
## Presolve removed 333701 rows and 151600 columns
## Presolve time: 116.59s
## Presolved: 30505 rows, 30300 columns, 129690 nonzeros
## Variable types: 200 continuous, 30100 integer (30100 binary)
## Found heuristic solution: objective 64.0000000
## Presolved: 30505 rows, 30300 columns, 129690 nonzeros
##
## Presolve removed 30103 rows and 11814 columns
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    0.0000000e+00   2.865759e+01   5.535568e+08    122s
##      177    2.0000000e+01   0.000000e+00   0.000000e+00    122s
##      177    2.0000000e+01   0.000000e+00   0.000000e+00    122s
##
## Root relaxation: objective 2.000000e+01, 177 iterations, 1.37 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
## *    0     0               0      20.0000000   20.00000  0.00%     -  122s
##
## Explored 0 nodes (179 simplex iterations) in 122.19 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 2.000000000000e+01, best bound 2.000000000000e+01, gap 0.0%

## Warning in validityMethod(object): object@space.held contains values less
## than 0, some species are really poorly represented
```

```r
# show summary
summary(sim_mrs_space3)
```

```
##   Run_Number Status Score Cost Planning_Units Connectivity_Total
```

```
## 1           1 MANUAL    20   20             20                 220
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1              28               161               31
##   Connectivity_In_Fraction
## 1                0.1272727
```

```r
# plot prioritisation
plot(sim_mrs_space3, 1)
```

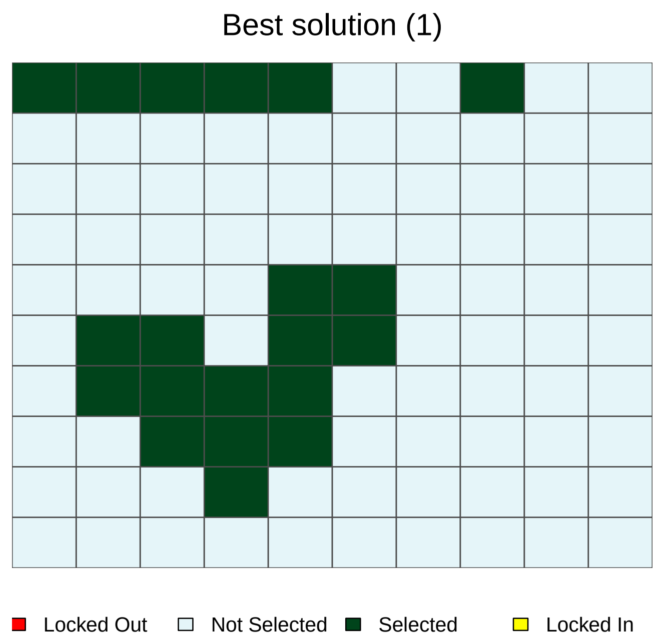


Figure 17 A multi-species prioritisation for the uniformly, normally, and bimodally distributed species generated using amount-based targets (20%) and space-based targets (85%). This priorititisation was generated to be robust against low occupancy probabilities, by explicitly using the probability of occupancy data when deriving a solution. See Figure 12 caption for conventions.

```r
# difference between prioritisations based on unreliable and reliable formulation
plot(sim_mrs_space3, sim_mrs_space, 1, 1, main='Difference between solutions')
```

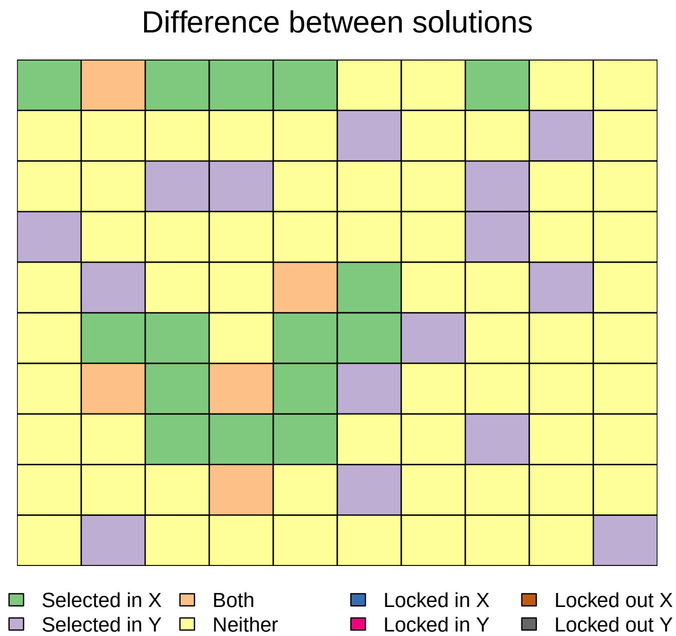


Figure 18 Difference between two multi-species prioritisations. See Figure 7 caption for conventions.

An additional planning unit was selected using the reliable formulation. The prioritisation based on the unreliable formulation had 20 planning units, but the prioritisation based on the reliable formlation has 20 planning units. This differnce occurs because the reliable formulation needs to ensure that all selected planning units with a low chance of being occupied have a suitable backup planning unit. While the reliable formulation can deliver more robust prioritisations, it takes much longer to solve conservation planning problems expressed using this formulation than the unreliable formulation. As a consequence, the reliable formulation is only feasible for particularly small problems, such as those involving few features and less than several hundred planning units.

Fragmentation

Fragmentation is an important consideration in real-world planning situations. Up until now, we haven't considered the effects of fragmentation on the viability of the prioritisation. As a consequence, our prioritisations have tended to contain planning units without any neighbours. We can use the `BLM` parameter to penalise fragmented solutions.

Let's generate a new prioritisation that heavily penalises fragmentation. Here, we will update the `sim_mrs_amount` object with `BLM` of 100.

```
# update prioritisation
sim_mrs_amount_blm <- update(sim_mrs_amount, BLM=100)
```

```
## Warning for adding variables: zero or small (< 1e-13) coefficients, ignored
## Optimize a model with 30666 rows, 30280 columns and 120720 nonzeros
## Coefficient statistics:
##   Matrix range    [6e-05, 8e+01]
##   Objective range [1e+02, 4e+02]
```

```
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 4e+02]
## Found heuristic solution: objective 4299
## Presolve removed 0 rows and 0 columns (presolve time = 5s) ...
## Presolve time: 5.56s
## Presolved: 30666 rows, 30280 columns, 120720 nonzeros
## Variable types: 0 continuous, 30280 integer (30280 binary)
## Presolved: 30666 rows, 30280 columns, 120720 nonzeros
##
## Presolve removed 30666 rows and 30280 columns
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    2.2100000e+04   0.000000e+00   4.010000e+04      7s
##     2198    1.1920905e+03   0.000000e+00   2.081845e+05     10s
##     5768    6.4673127e+02   0.000000e+00   2.068700e+05     15s
##     7117    4.6444444e+02   0.000000e+00   0.000000e+00     18s
##     7117    4.6444444e+02   0.000000e+00   0.000000e+00     18s
##
## Root relaxation: objective 4.644444e+02, 7117 iterations, 12.57 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0  464.44444    0 1725 4299.00000  464.44444  89.2%     -   18s
## H    0     0                    1953.0000000  464.44444  76.2%     -   46s
##      0     0  523.81679    0 1561 1953.00000  523.81679  73.2%     -   49s
##      0     0  523.81679    0 1560 1953.00000  523.81679  73.2%     -   50s
##      0     0  571.01549    0 1811 1953.00000  571.01549  70.8%     -   52s
##      0     0  571.01549    0 1827 1953.00000  571.01549  70.8%     -   53s
##      0     0  571.01549    0 1832 1953.00000  571.01549  70.8%     -   55s
##      0     0  571.01549    0 1831 1953.00000  571.01549  70.8%     -   56s
##      0     0  571.01549    0 1676 1953.00000  571.01549  70.8%     -   61s
##      0     0  571.01549    0 1684 1953.00000  571.01549  70.8%     -   61s
##      0     0  571.01549    0 1636 1953.00000  571.01549  70.8%     -   64s
##      0     0  571.01549    0 1651 1953.00000  571.01549  70.8%     -   65s
##      0     0  571.01549    0 1625 1953.00000  571.01549  70.8%     -   69s
##      0     0  571.01549    0 1631 1953.00000  571.01549  70.8%     -   70s
##      0     0  571.01549    0 1606 1953.00000  571.01549  70.8%     -   73s
##      0     0  571.01549    0 1611 1953.00000  571.01549  70.8%     -   74s
##      0     0  571.01549    0 1556 1953.00000  571.01549  70.8%     -   78s
##      0     0  571.01549    0 1559 1953.00000  571.01549  70.8%     -   78s
##      0     0  571.01549    0 1561 1953.00000  571.01549  70.8%     -   81s
##      0     0  571.01549    0 1562 1953.00000  571.01549  70.8%     -   81s
##      0     0  571.01549    0 1584 1953.00000  571.01549  70.8%     -   85s
##      0     0  571.01549    0 1539 1953.00000  571.01549  70.8%     -   87s
## H    0     0                    1435.0000000  571.01549  60.2%     -  100s
```

```
## H    0     0                        1332.0000000  571.01549  57.1%     -  111s
##      0     2  572.12289    0 1506 1332.00000  572.12289  57.0%     -  111s
##      8    10  645.00000    8 1030 1332.00000  575.86532  56.8%  1383  115s
##     12    14  716.12179   12 1110 1332.00000  575.86532  56.8%  1494  120s
##     20    22  994.33308   20  647 1332.00000  575.86532  56.8%  1257  125s
## H   22    22                        1227.0000000  575.86532  53.1%  1170  126s
## H   28    28                        1121.0000000  575.86532  48.6%   984  129s
##     35    35 1100.00000   33  145 1121.00000  575.86532  48.6%   821  130s
## *   41    33              38    1120.0000000  575.86532  48.6%   702  130s
##     52    42  747.11864   10 1265 1120.00000  585.46465  47.7%   721  135s
##     60    50  933.63636   18  910 1120.00000  585.46465  47.7%   710  141s
##     80    59     cutoff   29       1120.00000  585.46465  47.7%   611  145s
##     97    74  866.80851   17  979 1120.00000  594.82736  46.9%   597  150s
##    118    87  729.05018    6 1321 1120.00000  604.26637  46.0%   585  155s
##    124    93  869.00990   12  955 1120.00000  604.26637  46.0%   609  160s
##    135   100 1102.75862   20  571 1120.00000  604.26637  46.0%   616  165s
##    141   102  702.99776    7 1479 1120.00000  613.10042  45.3%   706  175s
##    155   116  906.11111   21  643 1120.00000  613.10042  45.3%   694  180s
##    169   123  712.59259    7  998 1120.00000  621.72786  44.5%   694  185s
##    185   139  946.47059   20  636 1120.00000  621.72786  44.5%   677  190s
##    202   151  797.19334    9 1063 1120.00000  625.00000  44.2%   666  195s
##    214   157 1113.54839   18  619 1120.00000  625.00000  44.2%   664  200s
##    236   160  724.08163    9 1022 1120.00000  641.89958  42.7%   628  205s
##    242   166  882.79070   15  995 1120.00000  641.89958  42.7%   647  210s
##    254   171  744.59580    3 1572 1120.00000  650.00000  42.0%   671  219s
##    255   172  802.68457    4 1471 1120.00000  650.00000  42.0%   671  220s
##    259   176 1063.92292    8  589 1120.00000  650.00000  42.0%   679  225s
##    266   175     cutoff   12       1120.00000  650.00000  42.0%   690  232s
##    278   179  822.43902   12  963 1120.00000  654.29758  41.6%   681  235s
## H  289   149                        1021.0000000  654.29758  35.9%   676  240s
## H  353   149                        1020.0000000  654.29758  35.9%   561  243s
##    357   148  734.68466    6 1300 1020.00000  655.58472  35.7%   563  245s
##    376   160  693.74630    3 1589 1020.00000  658.46154  35.4%   557  250s
##    391   175  876.36364   18  983 1020.00000  658.46154  35.4%   562  255s
##    411   182  887.56757   11  889 1020.00000  660.85511  35.2%   559  261s
##    417   188  924.76190   13  580 1020.00000  660.85511  35.2%   572  265s
##    428   190  792.16363    2 1554 1020.00000  663.02381  35.0%   577  271s
##    454   208  752.12291    8 1222 1020.00000  663.90244  34.9%   562  275s
##    460   214  931.32075   14  948 1020.00000  663.90244  34.9%   568  281s
##    484   227  945.00000   20  279 1020.00000  671.06383  34.2%   554  285s
##    500   235  864.59459    7 1559 1020.00000  678.06452  33.5%   553  290s
##    511   245  834.28571   16 1539 1020.00000  678.06452  33.5%   547  314s
##    513   246  691.48733    4 1725 1020.00000  678.06452  33.5%   545  356s
##    514   247  920.00000   15 1642 1020.00000  678.06452  33.5%   544  361s
##    516   248  842.44898   15 1500 1020.00000  678.06452  33.5%   542  387s
##    518   250 1001.81818   17 1555 1020.00000  678.06452  33.5%   540  394s
##    519   250  898.94737   16 1594 1020.00000  678.06452  33.5%   539  396s
##    520   251  882.50000   12 1486 1020.00000  678.06452  33.5%   538  407s
```

```
##     521    252   803.27125    6 1563 1020.00000  678.06452  33.5%    537  410s
##     522    252   737.39130   10 1562 1020.00000  678.06452  33.5%    536  415s
##     524    254   943.40426   15 1571 1020.00000  678.06452  33.5%    534  423s
##     525    254   898.12500   17 1532 1020.00000  678.06452  33.5%    533  427s
##     526    255   717.89474   10 1483 1020.00000  678.06452  33.5%    532  438s
##     527    256   898.94737   16 1500 1020.00000  678.06452  33.5%    531  440s
##     528    256   705.00031    4 1439 1020.00000  678.06452  33.5%    530  450s
##     530    258   743.34999    8 1472 1020.00000  678.06452  33.5%    528  461s
##     532    259   718.75000    7 1444 1020.00000  678.06452  33.5%    526  466s
##     534    260   920.00000   15 1444 1020.00000  678.06452  33.5%    524  475s
##     535    263   678.06452   15 1329 1020.00000  678.06452  33.5%    692  483s
##     536    264   694.47740   15 1499 1020.00000  678.06452  33.5%    695  485s
##     539    266   787.84598   17 1565 1020.00000  678.06452  33.5%    706  490s
##     544    268  1013.54839   19  775 1020.00000  678.06452  33.5%    728  495s
##     553    267  1014.11765   24  342 1020.00000  678.06452  33.5%    727  500s
##     562    267   731.25950   18 1387 1020.00000  678.06452  33.5%    732  505s
##     563    268   758.61850   19 1629 1020.00000  678.06452  33.5%    747  513s
##     566    270   843.05389   20 1577 1020.00000  678.06452  33.5%    752  516s
##     570    272   910.90909   22  648 1020.00000  678.06452  33.5%    761  520s
## *   571    254                23      920.0000000  678.06452  26.3%    761  522s
##     577    250   898.78788   25 1369  920.00000  678.06452  26.3%    764  525s
##     581    251   816.94578   17 1534  920.00000  689.31692  25.1%    770  530s
##     583    252   851.84478   18 1511  920.00000  689.31692  25.1%    786  536s
##     587    252      cutoff   20       920.00000  689.31692  25.1%    791  540s
##     596    250   881.19152   20  719  920.00000  691.34181  24.9%    794  545s
##     607    249   826.25000   20  597  920.00000  710.96046  22.7%    802  553s
##     608    250   857.37717   21 1229  920.00000  710.96046  22.7%    812  558s
##     610    249   897.77778   22  577  920.00000  710.96046  22.7%    811  560s
##     612    248   892.72727   22  569  920.00000  710.96046  22.7%    817  565s
##     615    247      cutoff   24       920.00000  710.96046  22.7%    824  570s
##     622    248   790.00000   20  941  920.00000  715.05609  22.3%    830  576s
##     626    250   853.33333   22  640  920.00000  715.05609  22.3%    830  582s
##     627    249   916.55172   23  612  920.00000  715.05609  22.3%    832  585s
##     633    247   788.41438   18 1248  920.00000  731.83346  20.5%    834  590s
##     642    245   903.78378   20  810  920.00000  731.83346  20.5%    834  595s
##     658    240   827.58226   19 1348  920.00000  732.65856  20.4%    825  601s
##     664    238   862.22988   20 1144  920.00000  732.65856  20.4%    826  605s
##     676    237   891.42857   20 1011  920.00000  782.77632  14.9%    826  610s
##     703    228   909.47368   46  388  920.00000  782.77632  14.9%    805  615s
##     728    221   888.42105   22  933  920.00000  801.98120  12.8%    787  620s
##     732    218   840.82613   20 1367  920.00000  805.11605  12.5%    794  626s
##     751    211   905.24590   38 1325  920.00000  805.11605  12.5%    783  630s
##     853    177   905.24590  140 1323  920.00000  805.11605  12.5%    691  635s
##     880    167   828.68077   17 1337  920.00000  808.16731  12.2%    681  652s
##     883    167      cutoff   19       920.00000  808.16731  12.2%    686  655s
##     895    161   912.67370   24 1124  920.00000  812.30628  11.7%    689  660s
##     909    157   918.12293   38 2347  920.00000  812.30628  11.7%    686  669s
##     910    157   918.71776   39 1113  920.00000  812.30628  11.7%    687  671s
```

```
##     915   153  904.61538   23   617   920.00000  818.45124  11.0%   689  675s
##     942   142     cutoff   19         920.00000  819.16697  11.0%   675  680s
##     951   140  902.92683   20   904   920.00000  825.63973  10.3%   679  685s
##     956   138  903.87097   20   577   920.00000  828.87080  9.91%   683  691s
##     961   133  853.33333   21   581   920.00000  832.50000  9.51%   684  697s
##     962   133  904.61538   22   581   920.00000  832.50000  9.51%   686  701s
##     982   127  856.36364   21   624   920.00000  838.18182  8.89%   676  705s
##     993   122  893.68421   22   795   920.00000  843.52941  8.31%   677  710s
##    1000   117  867.82609   21   644   920.00000  850.43478  7.56%   682  716s
##    1001   117  906.36364   22   612   920.00000  850.43478  7.56%   684  720s
##    1011   110  916.55172   22   588   920.00000  853.33333  7.25%   683  726s
##    1034   103     cutoff   25         920.00000  853.33333  7.25%   672  730s
##    1051    94  905.46862   20  1317   920.00000  862.02517  6.30%   667  735s
##    1054    93  909.22095   23   763   920.00000  862.02517  6.30%   672  740s
##    1113    72  886.66667   20   639   920.00000  868.81512  5.56%   642  746s
##    1133    66  899.15360   26   800   920.00000  870.00000  5.43%   636  750s
##
## Cutting planes:
##   Gomory: 11
##   Zero half: 69
##
## Explored 1157 nodes (809290 simplex iterations) in 753.32 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 9.200000000000e+02, best bound 8.780000000000e+02, gap 4.5652%
```

```r
# show summary of prioritisation
summary(sim_mrs_amount_blm)
```

```
##   Run_Number Status Score Cost Planning_Units Connectivity_Total
## 1          1 MANUAL  1420   20             20                220
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1              35               171               14
##   Connectivity_In_Fraction
## 1                0.1590909
```

```r
# show amount held for each prioritisation
amount.held(sim_mrs_amount_blm)
```

```
##   uniform    normal   bimodal
## 1     0.2 0.2645832 0.3911267
```

```r
# show space held for each prioritisation
space.held(sim_mrs_amount_blm)
```

```
##   uniform (Space 1) normal (Space 1) bimodal (Space 1)
## 1         0.4545455        0.4517326         0.6539667
```

```r
# plot prioritisation
plot(sim_mrs_amount_blm, 1)
```

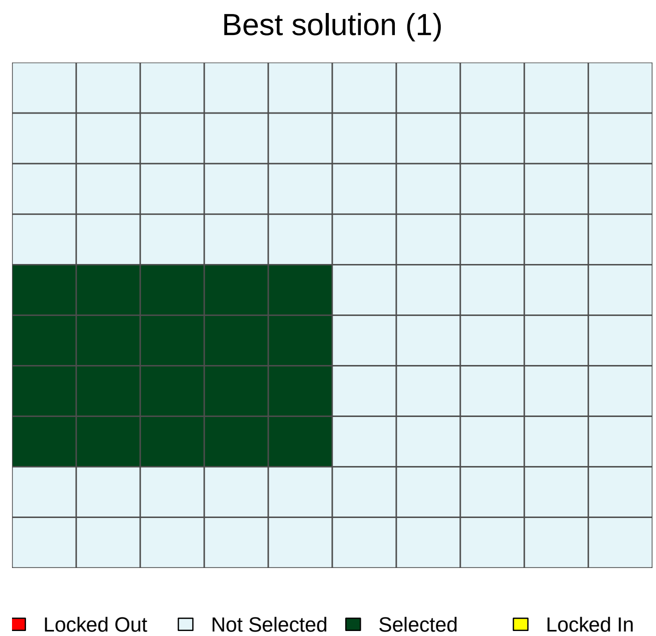


Figure 19 A multi-species prioritisation for the uniformly, normally, and bimodally distributed species generated using only amount-based targets (20%). Additionally, this priorititisation was specified to have high connectivity, by using a high BLM parameter. See Figure 12 caption for conventions.

```r
# difference between the two prioritisations
plot(sim_mrs_amount_blm, sim_mrs_amount, 1, 1, main='Difference between solutions')
```

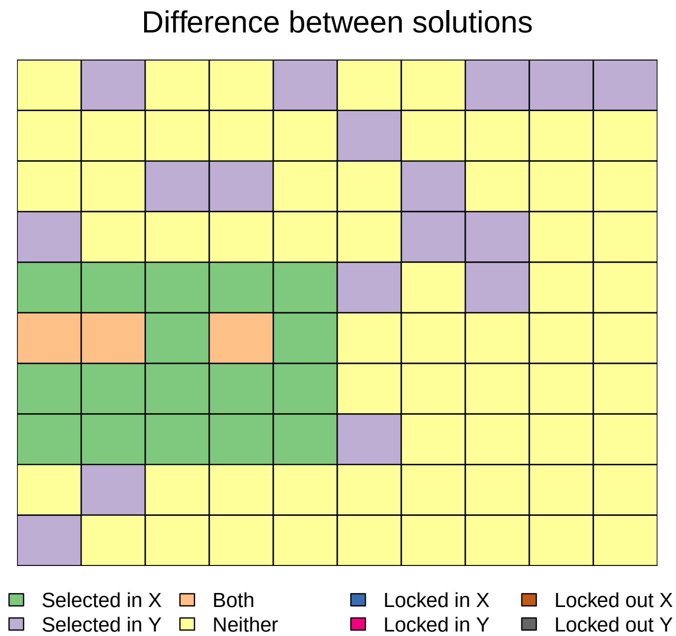


Figure 20 Difference between two multi-species prioritisations. See Figure 7 caption for conventions.

Here we can see that the prioritisation generated using a `BLM` parameter of 100 is much more clustered than the prioritisation generated using a `BLM` of 0. In practice, conservation planners will need to try a variety of `BLM` parameters to find a suitable prioritisation.

Complex simulated species

Data

In the previous examples, we have only used Euclidean distances to determine how much of an attribute space is sampled by a prioritisation. However, Euclidean distances can be poor measures of distance for multivariate, binary, or correlated variables (Faith *et al.* 1987). As a consequence this may lead to over- or under-estimates of the quality of a given solution.

The **rapr** R package provides a suite of distance metrics that can be used to calculate spatial representation (see `?AttributeSpace` for available metrics). To illustrate how using different distance metrics can affect the resulting solution, we will generate a new suite of prioritisations using different distance metrics.

First, we will simulate a new species and a three-dimensional attribute space. Note that unlike the previous examples, the attribute space will not be geographic space. Rather, each dimension in the attribute space will have values that map onto geographic space (eg. like climatic variables across the landscape). To add further complexity, we simulate their distributions using Gaussian processes.

```
# set seed for simulations
set.seed(500)

## simulate planning units
```

```r
sim_pus <- sim.pus(25L)

# simulate species
sim_gspp <- sim.species(sim_pus, model=RPgauss(), n=1, res=0.1)

# simulate space
sim_gspace <- sim.space(sim_pus, model=RMgauss(scale=3), d=3, res=0.1)
```

```
## ...
```

```r
# increment simulated space values by 100 so there are no negative values
# so we can investigate all distance metrics
sim_gspace <- sim_gspace + 100
```

```r
# generate RapUnsolved object containing data to generate prioritisations
sim_ru_gp <- rap(
    sim_pus, sim_gspp, sim_gspace,
    amount.target=0.2, space.target=0.85,
    n.demand.points=50L, kernel.method='hypervolume',
    include.geographic.space=FALSE, scale=FALSE, solve=FALSE
)
```

```
## Choosing repsperpoint=1500 (use a larger value for more accuracy.)
## Evaluating probability density...
## Building tree... done.
## Querying tree... 2.33918e-06  0.0233942  0.046786  0.0701778  0.0935696  0.116961  0.140353
## Finished evaluating probability density.
## Beginning volume calculation... done.
## Quantile requested: 0.20   obtained: 0.20
```

Let's visualise the species' distribution and the distribution of the attribute space across geographic space.

```r
# plot species distribution
plot(
    sim_gspp,
    main='Simulated species',
    col=colorRampPalette(c("#FFFFD9", "#EDF8B1", "#C7E9B4", "#7FCDBB",
        "#41B6C4", "#1D91C0", "#225EA8", "#253494", "#081D58"
    ))(100)
)
lines(sim_pus)
```

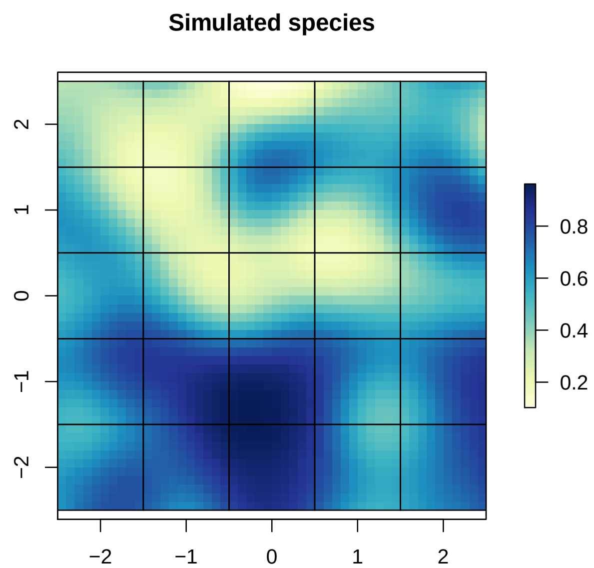


Figure 21 Distribution map of a species simulated using Gaussian prorceses. See Figure 2 caption for conventions.

```r
# plot distribution of each dimension in the attribute space across geographic space
plot(
    sim_gspace,
    main=c('Attribute space (d=1)', 'Attribute space (d=2)', 'Attribute space (d=3)'),
    addfun=function(){lines(sim_pus)},
    nc=3
)
```

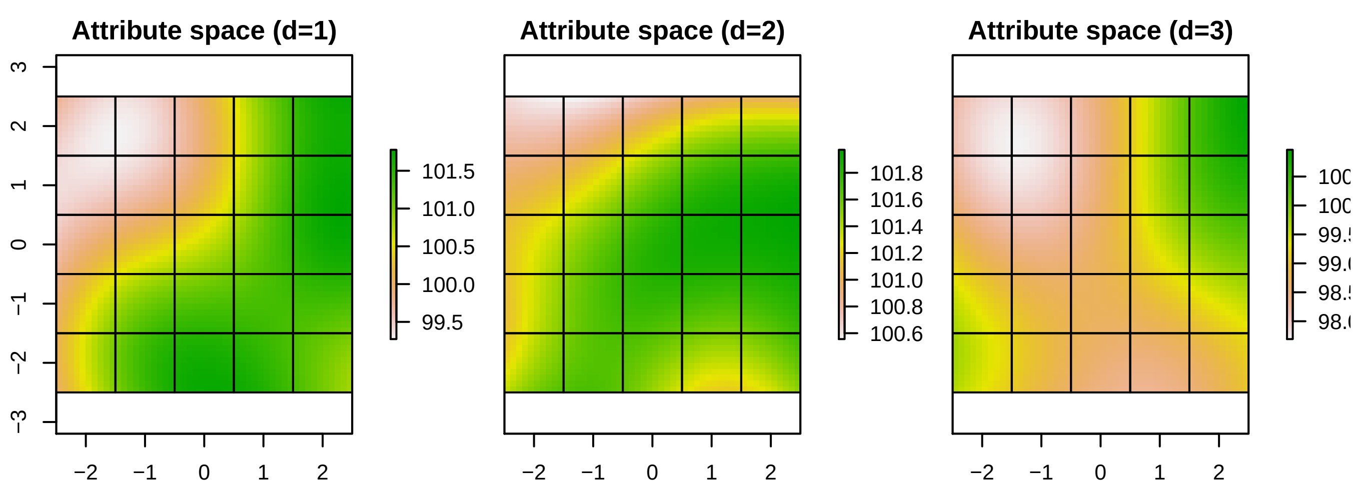


Figure 22 Distribution of spatial variables across the species' geographic range. These variables each represent a dimension of a three-dimensional attribute space.

Distance metrics

For each different distance metric, we will update the sim_gru object with the new distance metric, solve it, and store the solution in a list.

```
# create vector with distance metrics
dist.metrics <- c(
    'euclidean', 'bray', 'manhattan','gower',
    'canberra', 'mahalanobis',
    'jaccard', 'kulczynski'
)

# generate solutions
solutions <- list()
for (i in dist.metrics) {
    solutions[[i]] <- update(sim_ru_gp, distance.metric=i)
}
```

```
## Optimize a model with 1302 rows, 1275 columns and 5025 nonzeros
## Coefficient statistics:
##   Matrix range    [3e-01, 3e+04]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 5e+03]
## Presolve removed 248 rows and 143 columns
## Presolve time: 0.14s
## Presolved: 1054 rows, 1132 columns, 5329 nonzeros
## Variable types: 0 continuous, 1132 integer (1132 binary)
## Found heuristic solution: objective 19.0000000
## Presolved: 1054 rows, 1132 columns, 5329 nonzeros
##
## Presolve removed 1054 rows and 1132 columns
##
## Root relaxation: objective 4.583726e+00, 701 iterations, 0.04 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0    4.58373    0   69   19.00000    4.58373  75.9%     -    0s
## H    0     0                      5.0000000    4.58373  8.33%     -    0s
##
## Explored 0 nodes (701 simplex iterations) in 0.22 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 5.000000000000e+00, best bound 5.000000000000e+00, gap 0.0%
## Optimize a model with 1302 rows, 1275 columns and 5025 nonzeros
## Coefficient statistics:
```

```
##   Matrix range    [8e-06, 1e+00]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [4e-02, 3e+00]
## Presolve removed 185 rows and 134 columns
## Presolve time: 0.22s
## Presolved: 1117 rows, 1141 columns, 5590 nonzeros
## Variable types: 0 continuous, 1141 integer (1141 binary)
## Found heuristic solution: objective 19.0000000
## Presolve removed 22 rows and 0 columns
## Presolved: 1095 rows, 1141 columns, 5479 nonzeros
##
## Presolve removed 493 rows and 91 columns
##
## Root relaxation: objective 4.419702e+00, 718 iterations, 0.06 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0    4.41970    0   43   19.00000    4.41970  76.7%     -    0s
## H    0     0                       5.0000000    4.41970  11.6%     -    0s
##
## Explored 0 nodes (718 simplex iterations) in 0.32 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 5.000000000000e+00, best bound 5.000000000000e+00, gap 0.0%
## Optimize a model with 1302 rows, 1275 columns and 5025 nonzeros
## Coefficient statistics:
##   Matrix range    [3e-01, 9e+04]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 1e+04]
## Presolve removed 236 rows and 134 columns
## Presolve time: 0.14s
## Presolved: 1066 rows, 1141 columns, 5541 nonzeros
## Variable types: 0 continuous, 1141 integer (1141 binary)
## Found heuristic solution: objective 19.0000000
## Found heuristic solution: objective 18.0000000
## Presolved: 1066 rows, 1141 columns, 5541 nonzeros
##
## Presolve removed 1066 rows and 1141 columns
##
## Root relaxation: objective 4.422638e+00, 738 iterations, 0.04 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
```

```
##      0     0    4.42264    0   43   18.00000    4.42264  75.4%     -    0s
## H    0     0                       5.0000000    4.42264  11.5%     -    0s
##
## Explored 0 nodes (738 simplex iterations) in 0.22 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 5.000000000000e+00, best bound 5.000000000000e+00, gap 0.0%
## Optimize a model with 1302 rows, 1275 columns and 5025 nonzeros
## Coefficient statistics:
##   Matrix range    [5e-02, 1e+03]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 2e+02]
## Presolve removed 223 rows and 123 columns
## Presolve time: 0.15s
## Presolved: 1079 rows, 1152 columns, 5990 nonzeros
## Variable types: 0 continuous, 1152 integer (1152 binary)
## Found heuristic solution: objective 18.0000000
## Presolved: 1079 rows, 1152 columns, 5990 nonzeros
##
## Presolve removed 1079 rows and 1152 columns
##
## Root relaxation: objective 4.300502e+00, 706 iterations, 0.05 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0    4.30050    0   70   18.00000    4.30050  76.1%     -    0s
## H    0     0                       5.0000000    4.30050  14.0%     -    0s
##
## Explored 0 nodes (706 simplex iterations) in 0.24 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 5.000000000000e+00, best bound 5.000000000000e+00, gap 0.0%
## Optimize a model with 1302 rows, 1275 columns and 5025 nonzeros
## Coefficient statistics:
##   Matrix range    [7e-05, 2e+00]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [3e-01, 3e+00]
## Presolve removed 230 rows and 134 columns
## Presolve time: 0.16s
## Presolved: 1072 rows, 1141 columns, 5485 nonzeros
## Variable types: 0 continuous, 1141 integer (1141 binary)
## Found heuristic solution: objective 19.0000000
## Presolve removed 3 rows and 0 columns
```

```
## Presolved: 1069 rows, 1141 columns, 5465 nonzeros
##
## Presolve removed 1042 rows and 626 columns
##
## Root relaxation: objective 4.428373e+00, 571 iterations, 0.04 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0    4.42837    0   43   19.00000    4.42837  76.7%     -    0s
## H    0     0                       5.0000000    4.42837  11.4%     -    0s
##
## Explored 0 nodes (571 simplex iterations) in 0.23 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 5.000000000000e+00, best bound 5.000000000000e+00, gap 0.0%
## Optimize a model with 1302 rows, 1275 columns and 5025 nonzeros
## Coefficient statistics:
##   Matrix range    [3e-01, 5e+04]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 9e+03]
## Presolve removed 415 rows and 381 columns
## Presolve time: 0.12s
## Presolved: 887 rows, 894 columns, 6274 nonzeros
## Variable types: 0 continuous, 894 integer (894 binary)
## Found heuristic solution: objective 21.0000000
## Presolve removed 13 rows and 0 columns
## Presolved: 874 rows, 894 columns, 6203 nonzeros
##
## Presolve removed 874 rows and 894 columns
##
## Root relaxation: objective 9.752283e+00, 244 iterations, 0.03 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0    9.75228    0   23   21.00000    9.75228  53.6%     -    0s
## H    0     0                      10.0000000    9.75228  2.48%     -    0s
##
## Explored 0 nodes (244 simplex iterations) in 0.18 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 1.000000000000e+01, best bound 1.000000000000e+01, gap 0.0%
## Optimize a model with 1302 rows, 1275 columns and 5025 nonzeros
## Coefficient statistics:
```

```
##   Matrix range    [3e-05, 1e+00]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e-01, 3e+00]
## Presolve removed 207 rows and 134 columns
## Presolve time: 0.18s
## Presolved: 1095 rows, 1141 columns, 5597 nonzeros
## Variable types: 0 continuous, 1141 integer (1141 binary)
## Found heuristic solution: objective 19.0000000
## Presolve removed 13 rows and 0 columns
## Presolved: 1082 rows, 1141 columns, 5516 nonzeros
##
## Presolve removed 833 rows and 282 columns
##
## Root relaxation: objective 4.429824e+00, 591 iterations, 0.04 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0    4.42982    0   43   19.00000    4.42982  76.7%     -    0s
## H    0     0                      17.0000000    4.42982  73.9%     -    0s
## H    0     0                       5.0000000    4.42982  11.4%     -    0s
##
## Explored 0 nodes (591 simplex iterations) in 0.28 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 5.000000000000e+00, best bound 5.000000000000e+00, gap 0.0%
## Optimize a model with 1302 rows, 1275 columns and 5025 nonzeros
## Coefficient statistics:
##   Matrix range    [8e-06, 1e+00]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [3e-02, 3e+00]
## Presolve removed 184 rows and 134 columns
## Presolve time: 0.22s
## Presolved: 1118 rows, 1141 columns, 5819 nonzeros
## Variable types: 0 continuous, 1141 integer (1141 binary)
## Found heuristic solution: objective 19.0000000
## Presolve removed 16 rows and 0 columns
## Presolved: 1102 rows, 1141 columns, 5735 nonzeros
##
## Presolve removed 443 rows and 91 columns
##
## Root relaxation: objective 4.432050e+00, 692 iterations, 0.05 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
```

```
##
##      0     0    4.43205    0   43   19.00000    4.43205  76.7%     -    0s
## H    0     0                      5.0000000    4.43205  11.4%     -    0s
##
## Explored 0 nodes (692 simplex iterations) in 0.31 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 5.000000000000e+00, best bound 5.000000000000e+00, gap 0.0%
```

Now, let's plot the solutions to see how they differ.

```r
# set plotting window
par(mfrow=c(3,3), mar=c(0, 0, 4.1, 0))

## create plots showing the selected planning units (dark green)
for (i in seq_along(solutions)) {
    # plot i'th solution
    plot(
        sim_pus,
        main=dist.metrics[i],
        col=replace(
            rep('#ccece6',nrow(sim_pus@data)),
            which(selections(solutions[[i]])==1),
            '#00441b'
        ),
        axes=FALSE
    )
}

# reset plotting window
par(mfrow=c(1,1), mar=c(5.1, 4.1, 4.1, 2.1))
```

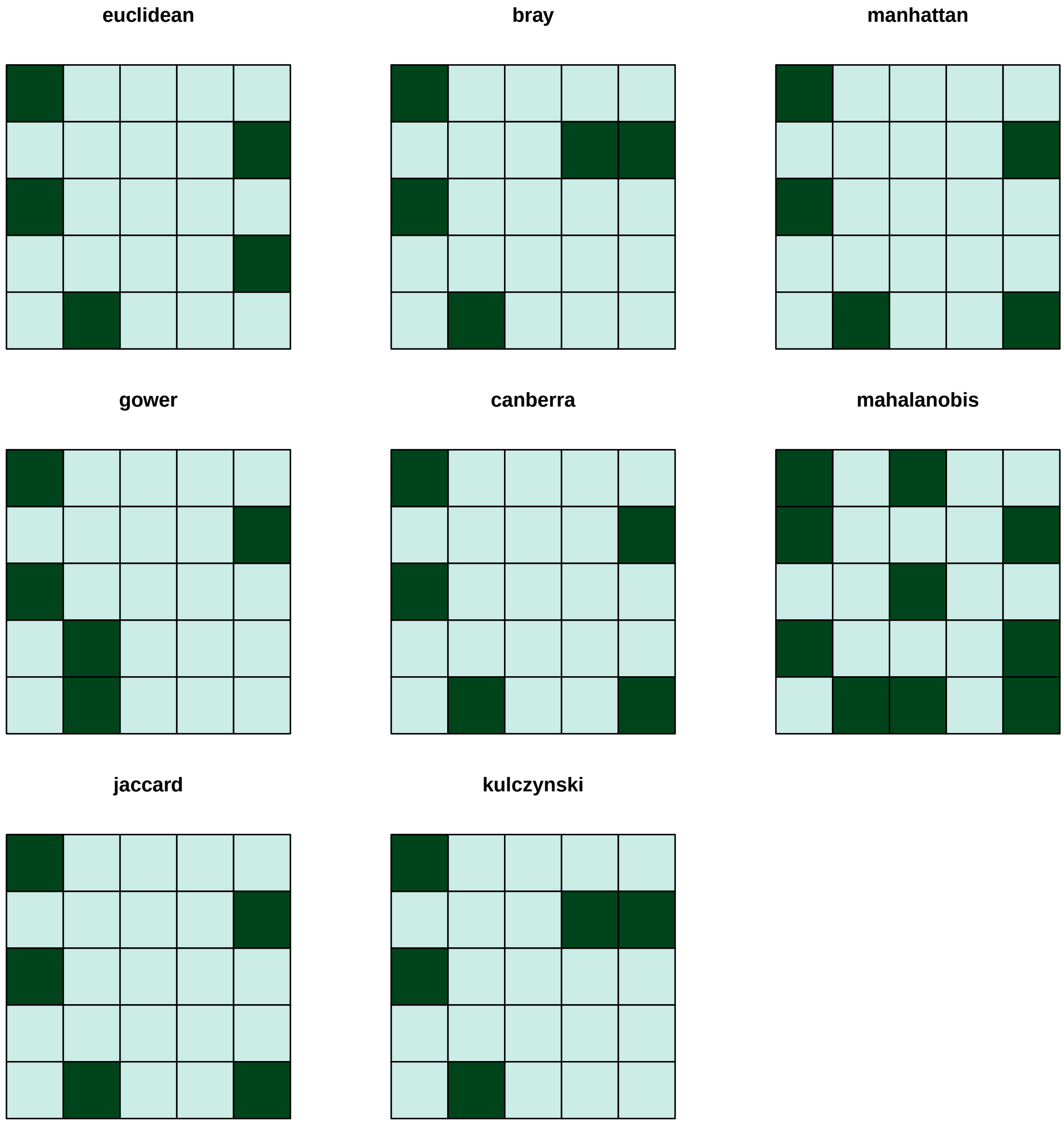


Figure 23 Prioritisations generated using different distance metrics. See Figure 12 for conventions.

It appears that main difference between the solutions is which planning units get selected in the bottom section of the study area. Some solutions tend to select a lot of planning units in this region (eg. Canberra, Euclidean, Gower, and Manhattan), whereas others select fewer planning units (eg. Bray-Curtis, Jaccard, and Kulczynski). Additionally, the Mahalanobis-based solution contains substantially more planning units than any other solution. Conservation planners should think carefully which distance metric is most appropriate for their attribute spaces. See the discussion section below for guidelines on selecting an appropriate distance metric.

Case-study examples

Overview

Here we will investigate how space-based targets can affect prioritisations using a more realistic dataset. We will generate prioritisations for the four bird species– blue-winged kookaburra, brown-backed honeyeater, brown falcon, pale-headed rosella–in Queensland, Australia. This region contains a broad range of different habitats–such as rainforests, woodlands, and deserts–making it ideal for this tutorial. First, we will generate a typical amount-based prioritisation that aims to capture 20% of the species' distributions. Then we will generate a prioritisation that also aims to secure populations in representative parts of the species' distributions in terms of their geographic location and their environmental heterogeneity. To do this we will generate a prioritisation using 20% amount-based targets and 85% space-based targets. Finally, we will compare these prioritisations to Australia's existing protected network.

Data

Survey data for the species were obtained from BirdLife Australia. The survey data was rarefied using a 100 km^2 grid, wherein the survey with the greatest number of repeat visits in each grid cell was chosen. To model the distribution of each species, environmental data were obtained at survey location (site). Specifically, climatic data (bio1, bio4, bio15, bio16, bio17) and classifications of the vegetation at the site were used. Occupancy-detection models (MacKenzie *et al.* 2002) were fit using Stan (Stan Development Team 2015) using manually tuned parameters (adapt deta=0.9, maximum treedepth=20, chains=4 , warmup iterations=1000, total iterations=1500) with five-fold cross-validation. In each replicate, data were partitioned into training and test sites. A full model was fit using quadratic terms for environmental variables in the site-component, and an intercept in the detection-component of the model. The full model was then subject to a step-wise backwards term deletion routine. Terms were retained when their inclusion resulted in a model with a greater area under the curve (AUC) value as calculated using the test data. Maps were generated for each species as an average of the predictions from the best model in each best replicate. To further improve the accuracy of these maps, areas well outside of the species' known distributions were set to 0. For each species, this was achieved by masking out biogeographic regions where the species was not observed, and regions that did not have a neighbouring planning unit where the species was observed. The maps were then resampled (10 km^2 resolution) and cropped to the study area. The resulting maps are stored in the `cs_spp` object.

```r
# load data
data(cs_spp)

# plot species' distributions
plot(cs_spp, main=c(
    "Blue-winged kookaburra", "Brown-backed honeyeater",
    "Brown falcon", "Pale-headed rosella"
))
```

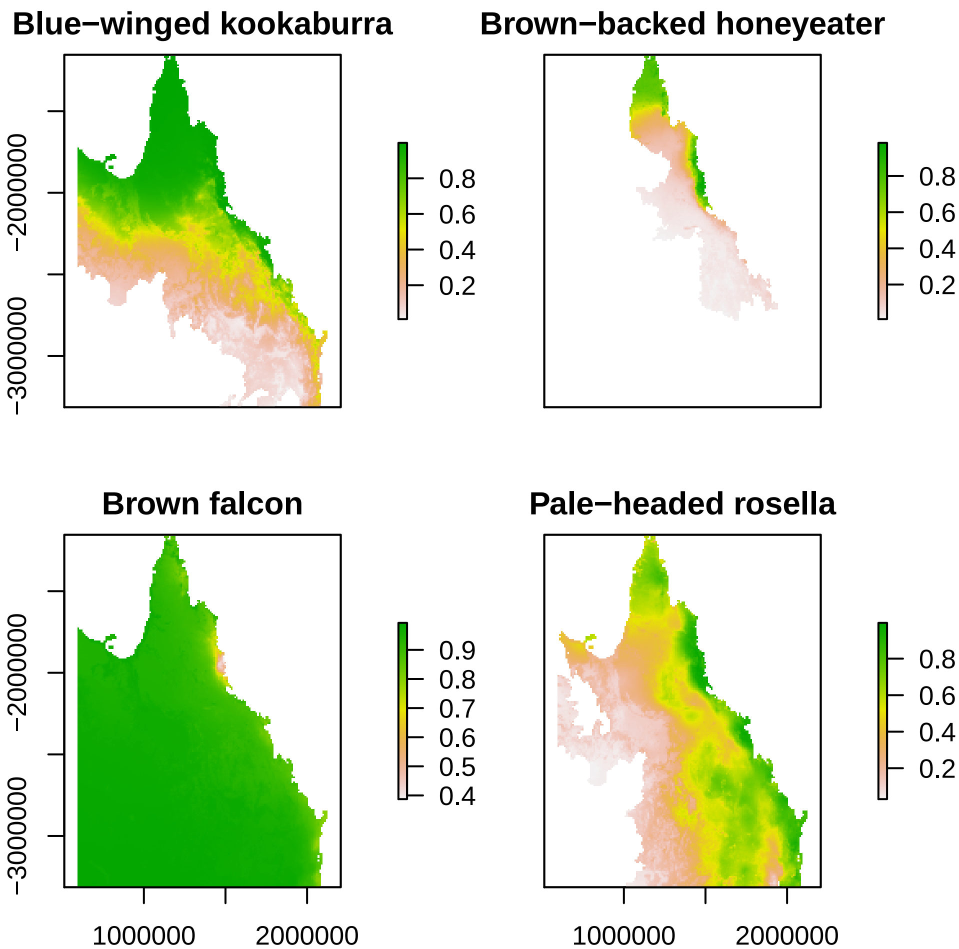


Figure 24 Distribution map for four Australian bird species. Pixel colours denote probability of occupancy.

Planning units (50km^2 resolution) were generated across Australia, and then clipped to the Queensland state borders and coastline. **Note that we are using excessively coarse planning units so that our examples will complete relatively quickly. In a real-world planning exercise, we would use much finer planning units.** To compare our prioritisations to Queensland's existing protected area network, this network was intersected with the planning units. Planning units with more than 50% of their area inside a protected area had their status set to 2 (following conventions in `Marxan`). Since we do not have cost data, the prioritisations will aim to select the minimum number of planning units required to meet the targets. The planning units are stored in the `cs_pu` object.

```
# load data
data(cs_pus)

## plot planning units
# convert SpatialPolygons to PolySet for quick plotting
cs_pus2 <- SpatialPolygons2PolySet(cs_pus)

# create vector of colours for planning units
```

```
# + light green: units not already inside reserve
# + yellow: units already inside reserve
cs_pus_cols <- rep('#c7e9c0', nrow(cs_pus@data))
cs_pus_cols[which(cs_pus$status==2)] <- 'yellow'

# set plotting window
par(mar=c(0.1, 0.1, 4.1, 0.1))

# plot polygons
PBSmapping::plotPolys(
    cs_pus2, col=cs_pus_cols, border='gray30',
    xlab='', ylab='', axes=FALSE,
    main='Case-study planning units',
    cex=1.8
)
```

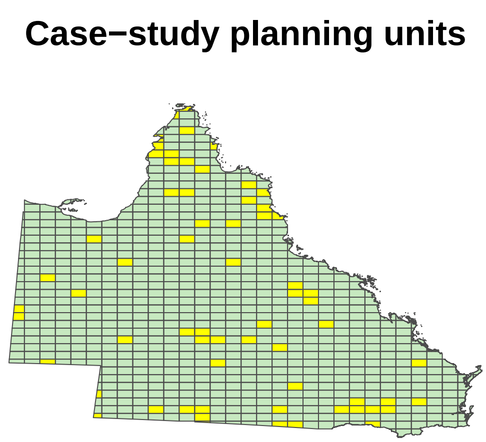


Figure 25 Planning units for the case-study examples. Polygons denote planning units. Yellow units have more than 50% of their area already in a reserve.

```
# reset plotting window
par(mar=c(5.1, 4.1, 4.1, 2.1))
```

To map the distribution of environmental conditions across the species' range, 21 bioclimatic layers were obtained. These layers were cropped to Australia and subject to detrended correspondence analysis to produce two new variables. These layers are stored in the `cs_space` object.

```
# load data
data(cs_space)

# plot variables
plot(cs_space, main=c('DC1', 'DC2'), legend=FALSE, axes=FALSE)
```

Figure 26 Broad-scale environmental variation across Australia. The variable DC1 describes the transition from wet and cool to dry and hot conditions. The variable DC2 describes the transition from wet and hot to dry and cool conditions.

Effectiveness of Australia's reserve network compared to optimal prioritisations

To simplify the process of formatting data and generating prioritisations, we can use the `rap` function. First, we will generate an amount-based prioritisation that aims to capture 20% of the rosella's range. We will use 50 demand points to map the geographic and environmental spaces. **Be warned, the examples hereafter can take 5-10 minutes to run.**

```
# make amount-based prioritisation,
# and ignore existing protected areas by discarding values in the
# status (third) column of the attribute table
cs_rs_amount <- rap(
    cs_pus[,-2], cs_spp, cs_space,
  amount.target=0.2, space.target=NA, n.demand.points=50L,
  include.geographic.space=TRUE, formulation='unreliable',
  solve=FALSE
)
```

```
## Warning in (function (pus, species, spaces = NULL, amount.target = 0.2, :
## argument to pus does not have a 'status' column, creating default with all
## status=0L
```

```
# threshold probabilities to 0.1 for space calculations
cs_rs_amount <- prob.subset(cs_rs_amount, species=1:4, threshold=rep(0.1,4))
```

```
# generate prioritisation
cs_rs_amount <- solve(cs_rs_amount)
```

```
## Optimize a model with 4 rows, 762 columns and 2001 nonzeros
## Coefficient statistics:
##   Matrix range    [3e+02, 2e+04]
```

```
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [2e+05, 3e+06]
## Found heuristic solution: objective 176
## Presolve time: 0.01s
## Presolved: 4 rows, 762 columns, 2001 nonzeros
## Variable types: 0 continuous, 762 integer (762 binary)
## Presolved: 4 rows, 762 columns, 2001 nonzeros
##
##
## Root relaxation: objective 1.359167e+02, 832 iterations, 0.01 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0  135.91668    0    4  176.00000  135.91668  22.8%     -    0s
## H    0     0                     136.0000000  135.91668  0.06%     -    0s
##
## Explored 0 nodes (832 simplex iterations) in 0.03 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 1.360000000000e+02, best bound 1.360000000000e+02, gap 0.0%
```

```
## Warning in validityMethod(object): object@space.held contains values less
## than 0, some species are really poorly represented
```

```r
# show summary
summary(cs_rs_amount)
```

```
##   Run_Number Status Score Cost Planning_Units Connectivity_Total
## 1          1 MANUAL   136  136            136           98882414
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1         9636021          81120163          8126230
##   Connectivity_In_Fraction
## 1               0.09744929
```

```r
# plot prioritisation
plot(cs_rs_amount, 1)
```

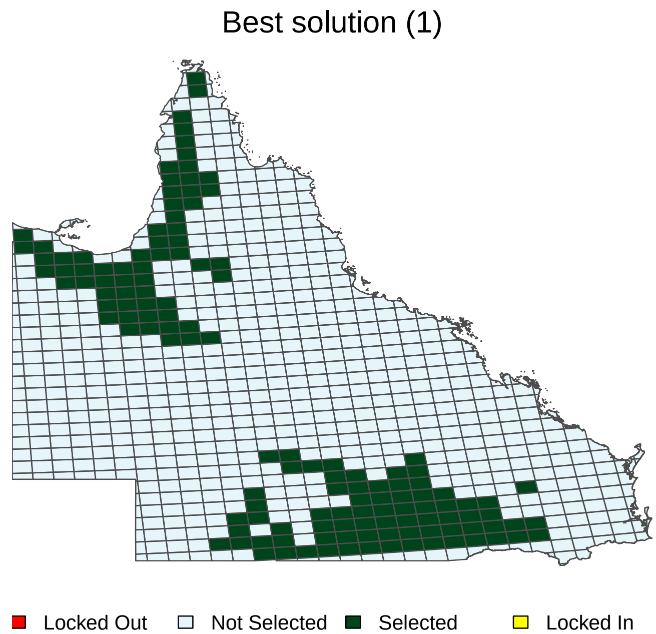


Figure 27 Multi-species prioritisation generated for four bird species using amount-based targets (20%). See Figure 12 captions for conventions.

We can also see how well the prioritisation secures the species' distributions in the geographic and environmental attribute spaces.

```
# plot prioritisation in geographic attribute space
p1 <- space.plot(cs_rs_amount, 1, 2, main='Blue-winged kookaburra')
p2 <- space.plot(cs_rs_amount, 2, 2, main='Brown-backed honeyeater')
p3 <- space.plot(cs_rs_amount, 3, 2, main='Brown falcon')
p4 <- space.plot(cs_rs_amount, 4, 2, main='Pale-headed rosella')
gridExtra::grid.arrange(p1, p2, p3, p4, ncol=2)
```

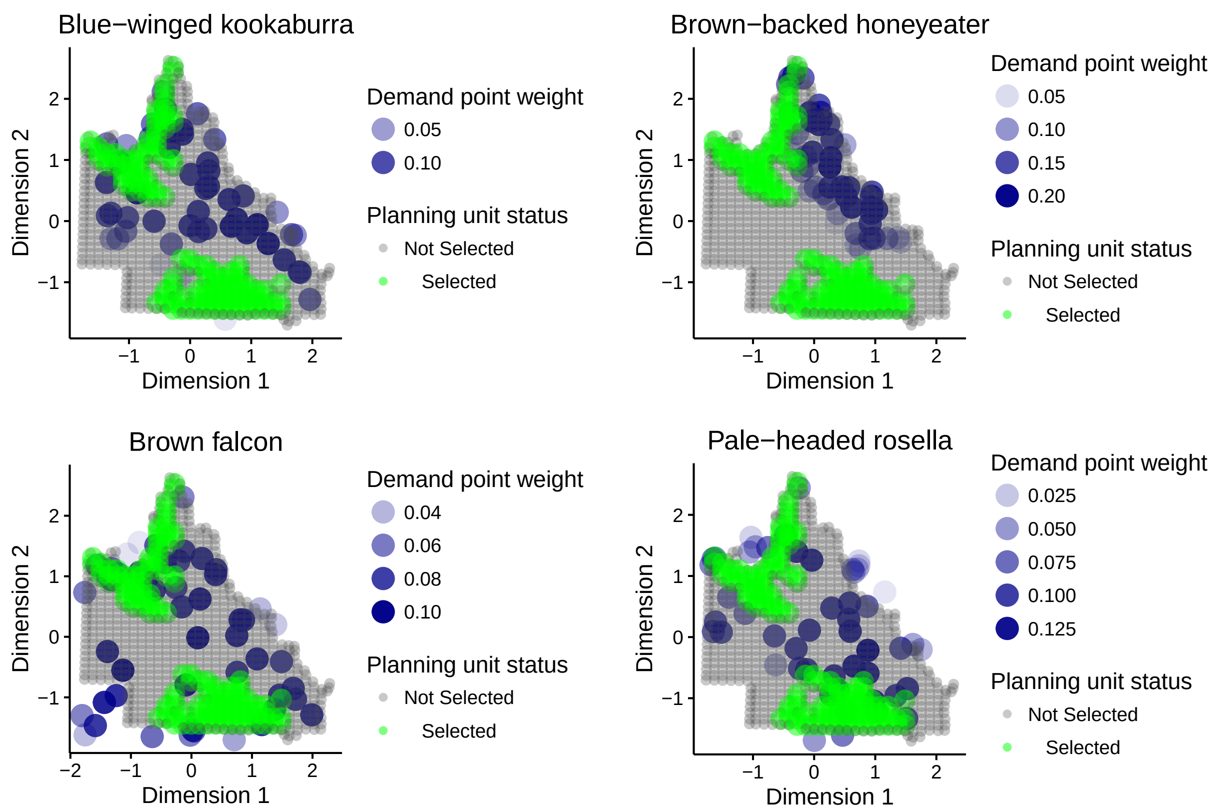


Figure 28 Distribution of amount-based prioritisation in the geographic attribute space. Points denote combinations of environmental conditions. Green and grey points represent planning unit selected for and not selected for prioritisation (respectively). Blue points denote demand points, and their size indicates their weighting.

```
# plot prioritisation in environmental attribute space
p1 <- space.plot(cs_rs_amount, 1, 1, main='Blue-winged kookaburra')
p2 <- space.plot(cs_rs_amount, 2, 1, main='Brown-backed honeyeater')
p3 <- space.plot(cs_rs_amount, 3, 1, main='Brown falcon')
p4 <- space.plot(cs_rs_amount, 4, 1, main='Pale-headed rosella')
gridExtra::grid.arrange(p1, p2, p3, p4, ncol=2)
```

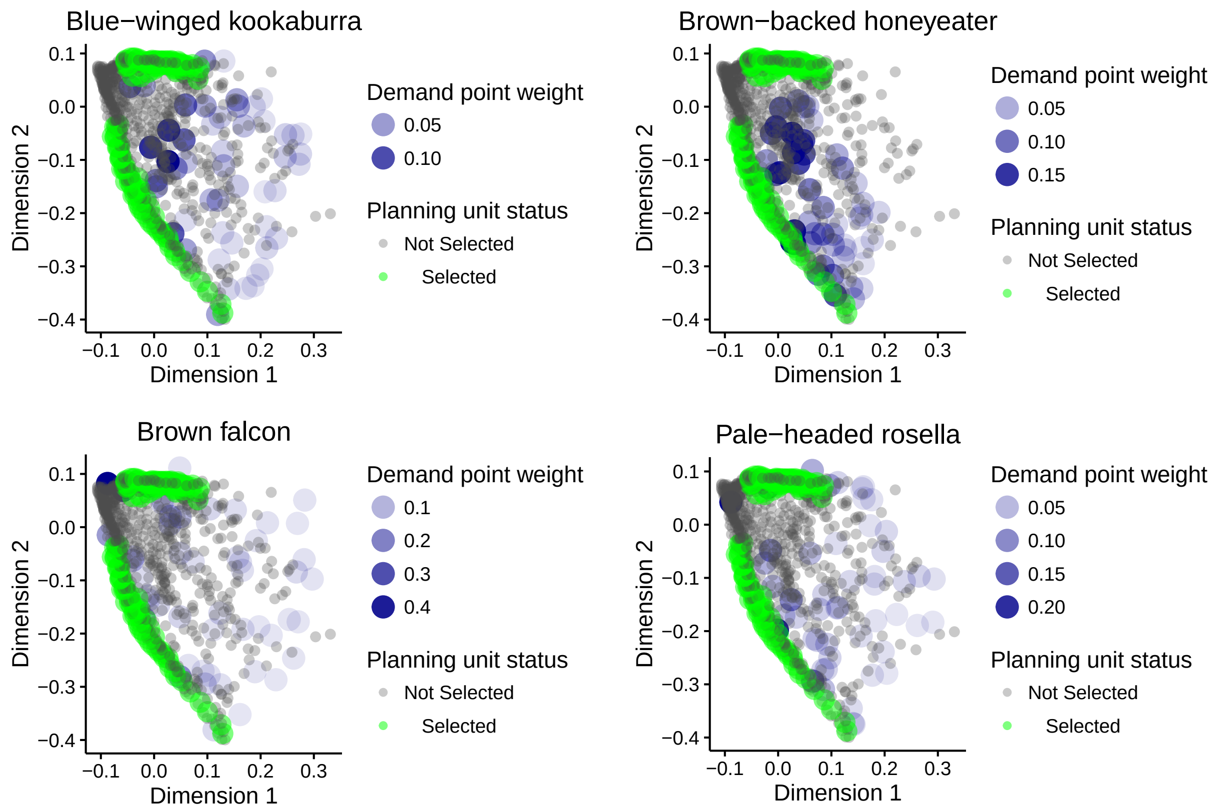


Figure 29 Distribution of amount-based prioritisation in the environmental attribute space. See Figure 28 caption for conventions.

Next, let's generate a prioritisation using amount- and space-based targets. This prioritisation will secure 50% of the species distribution in geographic and environmental space.

```
# make amount- and space-based prioritisation
cs_rs_space <- update(cs_rs_amount, space.target=0.5)
```

```
## Optimize a model with 200512 rows, 200862 columns and 802401 nonzeros
## Coefficient statistics:
##   Matrix range     [2e-11, 2e+04]
##   Objective range  [1e+00, 1e+00]
##   Bounds range     [1e+00, 1e+00]
##   RHS range        [3e-03, 3e+06]
## Warning: Model contains large matrix coefficient range
##          Consider reformulating model or setting NumericFocus parameter
##          to avoid numerical issues.
## Presolve removed 542 rows and 493 columns (presolve time = 5s) ...
## Presolve removed 587 rows and 493 columns (presolve time = 10s) ...
## Presolve removed 629 rows and 493 columns (presolve time = 15s) ...
## Presolve removed 657 rows and 493 columns (presolve time = 20s) ...
## Presolve removed 736 rows and 493 columns (presolve time = 25s) ...
```

```
## Presolve removed 811 rows and 493 columns (presolve time = 30s) ...
## Presolve removed 819 rows and 529 columns (presolve time = 35s) ...
## Presolve removed 819 rows and 529 columns (presolve time = 40s) ...
## Presolve removed 893 rows and 529 columns (presolve time = 45s) ...
## Presolve removed 893 rows and 529 columns (presolve time = 50s) ...
## Presolve removed 893 rows and 529 columns (presolve time = 55s) ...
## Presolve removed 893 rows and 529 columns (presolve time = 60s) ...
## Presolve removed 893 rows and 529 columns (presolve time = 65s) ...
## Presolve removed 893 rows and 529 columns (presolve time = 70s) ...
## Presolve removed 916 rows and 529 columns (presolve time = 75s) ...
## Presolve removed 937 rows and 529 columns (presolve time = 80s) ...
## Presolve removed 937 rows and 529 columns (presolve time = 85s) ...
## Presolve removed 937 rows and 529 columns (presolve time = 90s) ...
## Presolve removed 1045 rows and 529 columns (presolve time = 95s) ...
## Presolve removed 1083 rows and 529 columns (presolve time = 100s) ...
## Presolve removed 1113 rows and 529 columns (presolve time = 105s) ...
## Presolve removed 1147 rows and 529 columns (presolve time = 110s) ...
## Presolve removed 1147 rows and 529 columns (presolve time = 115s) ...
## Presolve removed 1153 rows and 529 columns (presolve time = 120s) ...
## Presolve removed 1153 rows and 529 columns (presolve time = 125s) ...
## Presolve removed 1153 rows and 529 columns (presolve time = 130s) ...
## Presolve removed 1191 rows and 529 columns (presolve time = 135s) ...
## Presolve removed 1191 rows and 529 columns (presolve time = 140s) ...
## Presolve removed 1191 rows and 529 columns (presolve time = 147s) ...
## Presolve removed 1191 rows and 529 columns (presolve time = 150s) ...
## Presolve removed 1191 rows and 529 columns (presolve time = 155s) ...
## Presolve removed 1191 rows and 529 columns (presolve time = 160s) ...
## Presolve removed 1191 rows and 529 columns (presolve time = 165s) ...
## Presolve removed 1191 rows and 529 columns (presolve time = 170s) ...
## Presolve removed 1191 rows and 529 columns (presolve time = 175s) ...
## Presolve removed 1191 rows and 529 columns (presolve time = 180s) ...
## Presolve removed 1191 rows and 967 columns (presolve time = 185s) ...
## Presolve removed 1191 rows and 967 columns (presolve time = 190s) ...
## Presolve removed 1191 rows and 967 columns (presolve time = 195s) ...
## Presolve removed 1191 rows and 967 columns (presolve time = 200s) ...
## Presolve removed 1591 rows and 967 columns (presolve time = 205s) ...
## Presolve removed 1635 rows and 967 columns (presolve time = 210s) ...
## Presolve removed 1717 rows and 967 columns (presolve time = 215s) ...
## Presolve removed 1762 rows and 967 columns (presolve time = 220s) ...
## Presolve removed 1779 rows and 967 columns (presolve time = 225s) ...
## Presolve removed 1806 rows and 967 columns (presolve time = 230s) ...
## Presolve removed 1806 rows and 967 columns (presolve time = 235s) ...
## Presolve removed 1806 rows and 967 columns (presolve time = 240s) ...
## Presolve removed 1806 rows and 967 columns (presolve time = 245s) ...
## Presolve removed 1806 rows and 967 columns (presolve time = 250s) ...
## Presolve removed 1806 rows and 967 columns
## Presolve time: 253.85s
## Presolved: 198706 rows, 199895 columns, 802062 nonzeros
```

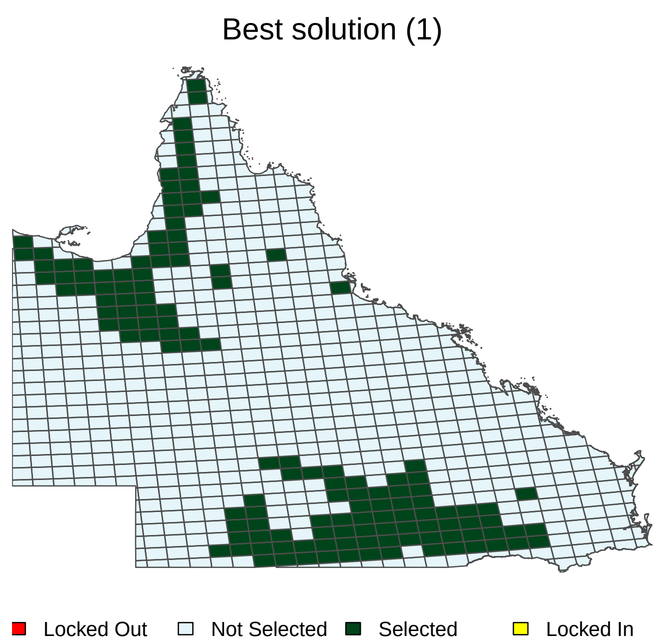
```
## Variable types: 0 continuous, 199895 integer (199895 binary)
## Found heuristic solution: objective 236.0000000
## Presolve removed 246 rows and 0 columns (presolve time = 5s) ...
## Presolve removed 248 rows and 0 columns (presolve time = 10s) ...
## Presolve removed 248 rows and 0 columns (presolve time = 15s) ...
## Presolve removed 248 rows and 0 columns
## Presolved: 198458 rows, 199895 columns, 801255 nonzeros
##
## Presolve removed 195493 rows and 9695 columns
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    0.0000000e+00   4.676020e+01   1.178365e+10    278s
##     4633    1.4108335e+02   0.000000e+00   1.312466e+03    280s
##     9008    1.3842417e+02   0.000000e+00   3.404555e+03    285s
##    13384    1.3765157e+02   0.000000e+00   1.108000e+03    290s
##    17272    1.3717381e+02   0.000000e+00   7.655796e+02    295s
##    22861    1.3704551e+02   0.000000e+00   2.470680e+03    300s
##    26020    1.3657798e+02   0.000000e+00   8.073856e+03    305s
##    30152    1.3625803e+02   0.000000e+00   2.848311e+03    310s
##    35255    1.3603035e+02   0.000000e+00   5.157527e+01    315s
##    39386    1.3599833e+02   0.000000e+00   3.159838e+00    320s
##    41032    1.3599204e+02   0.000000e+00   0.000000e+00    323s
##    41032    1.3599204e+02   0.000000e+00   0.000000e+00    324s
##
## Root relaxation: objective 1.359920e+02, 41032 iterations, 66.24 seconds
## Total elapsed time = 325.58s
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0  135.99204    0  189  236.00000  135.99204  42.4%     -  327s
## H    0     0                    137.0000000  135.99204  0.74%     -  329s
##
## Explored 0 nodes (53801 simplex iterations) in 329.43 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 1.370000000000e+02, best bound 1.360000000000e+02, gap 0.7299%
```

```r
# show summary
summary(cs_rs_space)
```

```
##   Run_Number Status Score Cost Planning_Units Connectivity_Total
## 1          1 MANUAL   137  137            137           98882414
##   Connectivity_In Connectivity_Edge Connectivity_Out
```

```
## 1          9738090          80918598          8225726
##   Connectivity_In_Fraction
## 1               0.09848151
```

```
# plot prioritisation
plot(cs_rs_space,1)
```




```
# plot prioritisation in geographic attribute space
p1 <- space.plot(cs_rs_space, 1, 2, main='Blue-winged kookaburra')
p2 <- space.plot(cs_rs_space, 2, 2, main='Brown-backed honeyeater')
p3 <- space.plot(cs_rs_space, 3, 2, main='Brown falcon')
p4 <- space.plot(cs_rs_space, 4, 2, main='Pale-headed rosella')
gridExtra::grid.arrange(p1, p2, p3, p4, ncol=2)
```

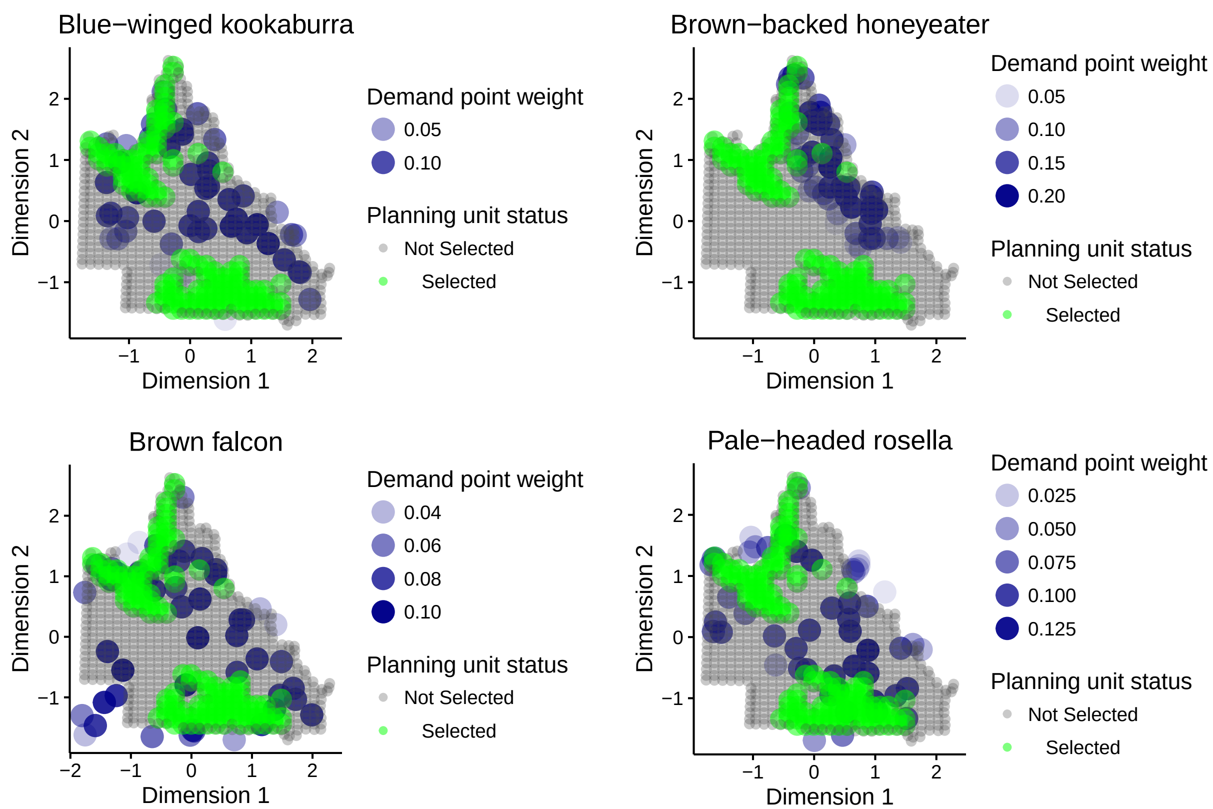


Figure 30 Distribution of the amount- and space-based prioritisation in the geographic attribute space. See Figure 28 caption for conventions.

```
# plot prioritisation in environmental attribute space
p1 <- space.plot(cs_rs_space, 1, 1, main='Blue-winged kookaburra')
p2 <- space.plot(cs_rs_space, 2, 1, main='Brown-backed honeyeater')
p3 <- space.plot(cs_rs_space, 3, 1, main='Brown falcon')
p4 <- space.plot(cs_rs_space, 4, 1, main='Pale-headed rosella')
gridExtra::grid.arrange(p1, p2, p3, p4, ncol=2)
```

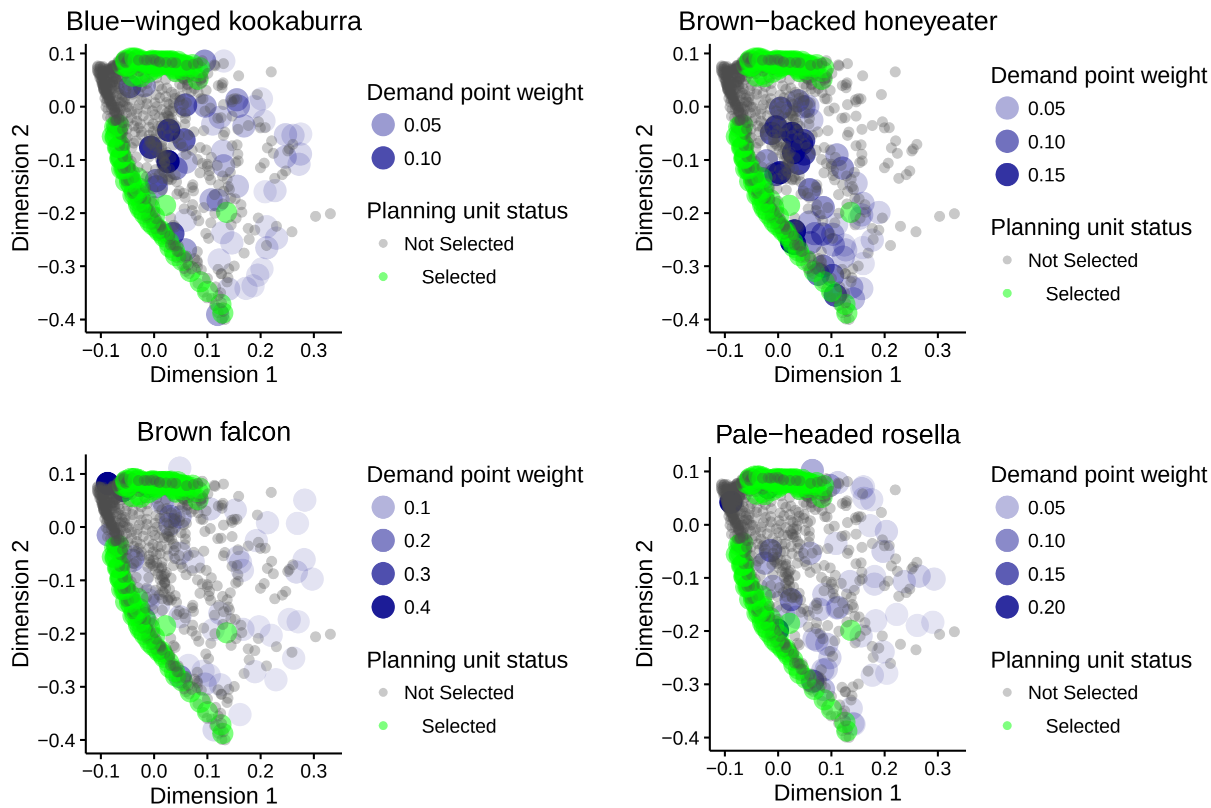


Figure 31 Distribution of the amount- and space-based prioritisation in the environmental attribute space. See Figure 28 caption for conventions.

Let's compare these prioritisations with Queensland's existing protected areas system. To do this, we can create update the `cs_rs_space` with manually specified solutions to create a `RapSolved` object to represent the Queensland's reserve network.

```
# generate vector with Australia's selections
aus_selections <- which(cs_pus$status>0)

# create new object with Australia's network
cs_rs_aus <- update(cs_rs_amount, b=aus_selections)
```

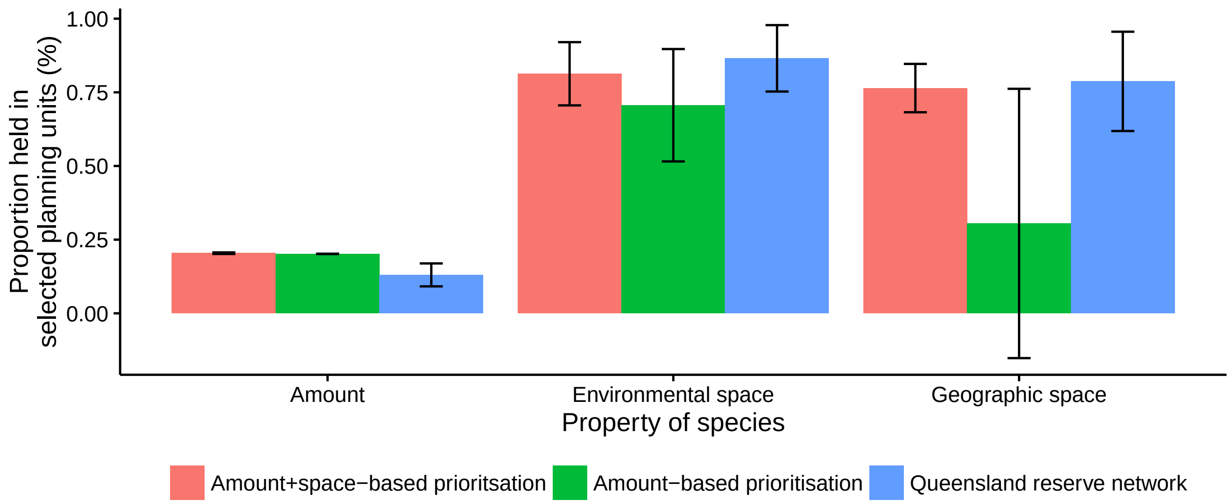
Now, let's plot the performance metrics for these prioritisations.

```
# define standard error function
se=function(x){sd(x,na.rm=TRUE)/sqrt(sum(!is.na(x)))}

# create a table to store the values for the 3 prioritisations
cs_results <- data.frame(
    name=rep(rep(c('Amount-based prioritisation',
        'Amount+space-based prioritsation', 'Queensland reserve network'),
        each=4),3),
```

```
    variable=rep(c('Amount', 'Geographic space', 'Environmental space'), each=12),
    species=colnames(amount.held(cs_rs_amount)),
    value=c(
        amount.held(cs_rs_amount)[1,], amount.held(cs_rs_space)[1,],
            amount.held(cs_rs_aus)[1,],
        space.held(cs_rs_amount, space=2)[1,], space.held(cs_rs_space, space=2)[1,],
            space.held(cs_rs_aus, space=2)[1,],
        space.held(cs_rs_amount, space=1)[1,], space.held(cs_rs_space, space=1)[1,],
            space.held(cs_rs_aus, space=1)[1,]
    )
) %>% group_by(
    name,
    variable
) %>% summarise(
    mean=mean(value),
    se=se(value)
)

# plot the performance metrics
ggplot(aes(x=variable, y=mean, fill=name), data=cs_results) +
    geom_bar(position=position_dodge(0.9), stat='identity') +
    geom_errorbar(
        aes(ymin=mean-se, ymax=mean+se), position=position_dodge(0.9),
        width=0.2
    ) +
    xlab('Property of species') +
    ylab('Proportion held in\nselected planning units (%)') +
    scale_fill_discrete(
        name=''
    ) +
    theme_classic() +
    theme(legend.position='bottom',legend.direction='horizontal')
```



We can see that a greater number of planning units is needed to satisfy the space-based targets. The prioritisation generated using just amount-based targets has 136 planning units, and the prioritisations using amount-based and space-based targets has 137 targets. These results suggest that prioritisations generated using only amount-based targets can obtain a moderately representative sample of the species' geographic distribution and climatic niche.

Implications and future directions

The `rapr` R package provides a unified approach to reserve selection. This R package provides decision makers with the tools to generate prioritisations that secure both biodiversity patterns and processes. Additionally, the package contains funcionality to accommodate uncertainty in the distribution of features, and also identify suitably connected reserves. Both the simulated and case-study species suggest that conservation planning exercises need to explicitly consider biodiversity processes during the reserve selection process to capture them.

One of the key advantages of the `rapr R` package is that it is general enough that any spatial variation could be considered an attribute space, regardless of whether this variation is intrinsic or extrinsic to the feature(s). For example, advances in genomic fields produced high resolution data on genetic information (eg. amplified fragment length polymorphisms, AFLPs; single nucleotide polymorphisms, SNPs). By using geostatistical analysis (eg. generalised dissimilarity modelling GDMs and gradient forests; Ferrier *et al.* 2007; Ellis *et al.* 2012), this data has been used to generate maps describing the spatial distribution of genomic variation within a species (Thomassen *et al.* 2010; Fitzpatrick and Keller 2015). These maps in turn could be used to construct a genomic attribute space, and in turn, could be used to generate prioritisations that secure a representative sample of genomic variation within a species. However, because the problem formulations used in this package are so general, the tools in this package could be misused, and generate poor quality prioritisations.

The degree to which a prioritisation truly secures a representative sample of a feature depends on the attribute spaces and distribution of demand points chosen by the decision maker. Ultimately, the space-based targets are set as a proportion based on the distribution of the demand-points. As a consequence, if the decision maker uses an inappropriate set of spatial variables to construct an attribute space, or an inappropriate set of demand points, then the optimal solution based on this data will not actually be an effective prioritisation. We therefore stress that decision makers must carefully consider which biodiversity processes need to preserved in the prioritisation, and what spatial data can be used to map these processes. To assist in the selection of appropriate demand points, the `R` package provides several routines for generating demand points (see the `make.DemandPoints` function). These routines essentially use the distribution of a feature in the attribute space to define a polygon. Demand points are then generated as random points within the polygon. A kernel is then fit to the distribution of the feature in the space (using Blonder *et al.* 2014; Duong 2015), and the demand points are weighted based on the estimated density of the feature at the demand points' coordinates.

The `rapr R` package could be further extended to identify more effective prioritisations. First, the formulation of fragmentation used in this package may be too simplistic in some cases (eg. exercises involving multiple species with different dispersal capabilities), and more realistic measures of fragmentation (eg. those used in Zonation) could be used to identify more effective prioritisations. Second, the problem does not consider temporal dynamics. Here, conservation actions are assumed to be implemented simultaneously in all selected planning units and assumed to remain implemented for all time. As a consequence, this R package is not useful for scenarios where actions are implemented during multiple discrete periods in time (eg. actions are made adue to annual funding cycles), or scenarios involving threatening processes that vary across space and time (reviewed in Pressey *et al.* 2007). Future research may look into incorporating such elements into this `R` package.

To maximise the long-term persistence of biodiversity–the stated goal of conservation–decision makers need to identify prioritisations that preserve existing patterns of biodiversity and the processes that support them. To achieve this, conservation planners need a decision support tool that can explicitly accommodate biodiversity patterns and processes. Here, we developed the `rapr R` package to fill this void. By exploring the functiontality of this package using several simulated species, we found that including space-based targets can radically change a prioritisation for the simplest of species.

Acknowledgements

JOH is funded by an Australian Postgraduate Award (APA) scholarship. RAF has an Australian Research Council Future Fellowship. This work was supported by the Centre of Excellence for Environmental Decisions (CEED) and the Landscape Ecology and Conservation Group (LEC) at The University of Queensland.

References

Ball, I., Possingham, H., Watts, M. E. (2009) Marxan and relatives: software for spatial conservation prioritisation. In: *Spatial conservation prioritisation: Quantitative methods & computational tools* (eds A. Moilanen, K. A. Wilson, & H. Possingham) pp. 185–189 Oxford University Press, Oxford, UK.

Beyer, H. L., Dujardin, Y., Watts, M. E., Possingham, H. P. (2016) Solving conservation planning problems with integer linear programming. *Ecological Modelling.*

Blonder, B., Lamanna, C., Violle, C., Enquist, B. J. (2014) The n-dimensional hypervolume. *Global Ecology and Biogeography.* **23**, 595–609.

Carvalho, S. B., Brito, J. C., Crespo, E. J., Possingham, H. P. (2011) Incorporating evolutionary processes into conservation planning using species distribution data: a case study with the western Mediterranean herpetofauna. *Diversity & Distributions.* **17**, 408–421.

Ciarleglio, M., Wesley Barnes, J., Sarkar, S. (2009) ConsNet: new software for the selection of conservation area networks with spatial and multi-criteria analyses. *Ecography.* **32**, 205–209.

Cornuéjols, G., Nemhauser, G. L., Wolsey, L. A. (1990) The uncapacitated facility location problem. In: *Discrete Location Theory* (eds P. B. Mirchandani & R. L. Francis) pp. 119–171 Wiley, New York.

Crandall, K. A., Bininda-Emonds, O. R. P., Mace, G. M., Wayne, R. K. (2000) Considering evolutionary processes in conservation biology. *Trends in Ecology & Evolution.* **15**, 290–295.

Cui, T. T., Ouyang, Y. F., Shen, Z. J. M. (2010) Reliable facility location design under the risk of disruptions. *Operations Research.* **58**, 998–1011.

Duong, T. (2015) *ks: Kernel Smoothing, version 1.9.4.*

Ellis, N., Smith, S. J., Pitcher, C. R. (2012) Gradient forests: calculating importance gradients on physical predictors. *Ecology.* **93**, 156–168.

Faith, D. P. (2003) Environmental diversity (ED) as surrogate information for species-level biodiversity. *Ecography.* **26**, 374–379.

Faith, D. P., Walker, P. A. (1996) Environmental diversity: on the best-possible use of surrogate data for assessing the relative biodiversity of sets of areas. *Biodiversity & Conservation.* **5**, 399–415.

Faith, D., Minchin, P., Belbin, L. (1987) Compositional dissimilarity as a robust measure of ecological distance. *Vegetatio.* **69**, 57–68.

Ferrier, S., Manion, G., Elith, J., Richardson, K. (2007) Using generalized dissimilarity modelling to analyse and predict patterns of beta diversity in regional biodiversity assessment. *Diversity and Distributions.* **13**, 252–264.

Fitzpatrick, M. C., Keller, S. R. (2015) Ecological genomics meets community-level modelling of biodiversity: mapping the genomic landscape of current and future environmental adaptation. *Ecology Letters.* **18**, 1–16.

Hendry, A. P., Lohmann, L. G., Conti, E., Cracraft, J., Crandall, K. A., Faith, D. P., Hauser, C., Joly, C. A., Kogure, K., Larigauderie, A., Magallon, S., Moritz, C., Tillier, S., Zardoya, R., Prieur-Richard, A. H., Walther, B. A., Yahara, T., Donoghue, M. J. (2010) Evolutionary biology in biodiversity science, conservation, and policy: A call to action. *Evolution.* **64**, 1517–1528.

Hijmans, R. J., Cameron, S. E., Parra, J. L., Jones, P. G., Jarvis, A. (2005) Very high resolution interpolated climate surfaces for global land areas. *International Journal of Climatology.* **25**, 1965–1978.

Kleiman, D. G. (1989) Reintroduction of captive mammals for conservation. *BioScience.* **39**, 152–161.

Klein, C., Wilson, K., Watts, M., Stein, J., Berry, S., Carwardine, J., Smith, M. S., Mackey, B., Possingham, H. (2009) Incorporating ecological and evolutionary processes into continental-scale conservation planning. *Ecological applications.* **19**, 206–217.

MacKenzie, D. I., Nichols, J. D., Lachman, G. B., Droege, S., Andrew Royle, J., Langtimm, C. A. (2002) Estimating site occupancy rates when detection probabilities are less than one. *Ecology.* **83**, 2248–2255.

Mahalanobis, P. C. (1936) On the generalized distance in statistics. *Proceedings of the National Institute of Sciences, Calcutta.*

Margules, C. R., Pressey, R. L. (2000) Systematic conservation planning. *Nature.* **405**, 243–253.

McNeely, J. A. (1994) Protected areas for the 21st-century: Working to provide benefits to society. *Biodiversity and Conservation.* **3**, 390–405.

Moilanen, A. (2007) Landscape Zonation, benefit functions and target-based planning: unifying reserve selection strategies. *Biological Conservation.* **134**, 571–579.

Moritz, C. (1994) Defining evolutionarily significant units for conservation. *Trends in Ecology & Evolution.* **9**, 373–375.

Moritz, C. (1999) Conservation units and translocations: strategies for conserving evolutionary processes. *Hereditas.* **130**, 217–228.

Moritz, C. (2002) Strategies to protect biological diversity and the evolutionary processes that sustain it. *Systematic Biology.* **51**, 238–254.

Pressey, R. L., Cabeza, M., Watts, M. E., Cowling, R. M., Wilson, K. A. (2007) Conservation planning in a changing world. *Trends in Ecology & Evolution.* **22**, 583–592.

Sanderson, E. W., Segan, D. B., Watson, J. E. M. (2015) Global status of and prospects for protection of terrestrial geophysical diversity. *Conservation Biology.* **29**, 649–656.

Sherali, H., Alameddine, A. (1992) A new reformulation-linearization technique for bilinear programming problems. *Journal of Global Optimization.* **2**, 379–410.

Snyder, L. V., Daskin, M. S. (2005) Reliability models for facility location: the expected failure cost case. *Transportation Science.* **39**, 400–416.

Stan Development Team (2015) RStan: The R interface to Stan, version 2.8.0.

Thomassen, H. A., Cheviron, Z. A., Freedman, A. H., Harrigan, R. J., Wayne, R. K., Smith, T. B. (2010) Spatial modelling and landscape-level approaches for visualizing intra-specific variation. *Molecular Ecology.* **19**, 3532–3548.

Vos, P., Hogers, R., Bleeker, M., Reijans, M., Lee, T. van de, Hornes, M., Friters, A., Pot, J., Paleman, J., Kuiper, M., Zabeau, M. (1995) AFLP: a new technique for DNA fingerprinting. *Nucleic Acids Research.* **23**, 4407–4414.