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

20 November 2015

Abstract

A central aim in conservation is to maximise the long-term persistence of biodiversity. To achieve this, reserve networks are used to safeguard biodiversity patterns and processes. But resources are limited. 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 not well suited for preserving biodiversity processes. To fill this gap in the conservation planning toolbox, we developed the `rapr` R package. This R package uses two new novel formulations of the reserve selection problem to identify prioritisations. Here, we explore the functionality of this R package using several simulated species and a case-study planning exercise in Queensland, Australia. We show how explicitly considering representativeness can alter a prioritisation. In most cases, we found that only a few additional plannning units are required to acheive far better representativeness.

Contents

Introduction

The overarching aim of conservation is to maximise the long-term persistence of biodiversity (McNeely 1994; Margules and Pressey 2000). To acheive 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 (Moritz 2002) 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 biodiversity processes (Margules and Pressey 2000; Crandall *et al.* 2000). Ecological processes are required for biodiversity to persist over short time-scales. Examples of these processes include predator-prey interactions, pollination, and decomposition. Typically, they operate over small geographic domains, with exceptions such as mirgation and refugial habitats, and can be preserved using suitably large planning units (Ciarleglio *et al.* 2009) that represent discrete blocks of habitat (Klein *et al.* 2009). On the other hand, evolutionary processes are required for biodiversity to persist over long time-scales. Typically, they operate over large geographic domains. Adaptive evolutionary processes can be preserved by securing populations with different adaptations and/or along selection pressure gradients (eg. environmental gradients; Moritz 2002; Crandall *et al.* 2000; Rouget *et al.* 2003; Cowling *et al.* 2003). Neutral evolutionary processes can be be preserved by securing populations with different evolutionary histories and/or with limited gene flow (Moritz 2002; Carvalho *et al.* 2011; Ponce-Reyes *et al.* 2014). Due to advances in technology over the last few decades, a wealth of data on biodiversity processes has become freely avaliable to conservation planners. Yet this data is only rarely used in conservation planning exercises (Hendry *et al.* 2010).

Existing decision support tools focus primarily on preserving biodiversity patterns or processes–but not both. Many tools have been developed to preserve biodiversity patterns (eg. C-Plan (Pressey *et al.* 2009); ConsNet (Ciarleglio *et al.* 2009); Marxan (Ball *et al.* 2009); Zonation (Moilanen 2007)). They use targets (eg. Marxan) or weights (eg. Zonation) to identify prioritisations that contain an adequate amount of individuals or habitat for a set of species (features). To accomodate some information on biodiversity processes, conservation planners can split features into sub-features as a pre-processing step (Carvalho *et al.* 2011). However, this approach is limited because data on biodiversity processes is often hyper-dimensional and continuous (eg. bioclimatic data), and these problems cannot structure within and between sub-features (Faith and Walker 1996). On the other hand, very few tools have been developed with a specific focus on biodiversity processes. The DIVERSITY decision support tool (Faith 2003) uses continuous data to site reserves in places that

secure a representative sample of variation (eg. environmental variation). However, unlike tools that primarily focus on biodiversity patterns, this tool can only accomodate data on the distribution of a single feature. Additionally, due to the nature of its optimisation problem, this tool can often deliver prioritisations that are excssively fragmented and provides no options to avoid this.

Today, one of the key issues preventing decision makers from explicitly considering both biodiversity patterns and processes in the reserve seleciton process is the lack of a decision support tool that can accomodate data on both of them in a multi-species context. To fill this void, we present the `rapr` R package. This R package provides decision makers with the tools to identify, visaulise, and compare prioritisations. We develop two novel formulations of the reserve selection problem that use targets to ensure an adequate amount of a representative portion of habitat is preserved for each feature. Finally, we provide a tutorial showcasing the functionality of this R package using three 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. 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).

Each attribute is conceptualised as a space–an attribute space–and planning units are thought to occupy points 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 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. The degree to which a prioritisation represents a spatial attribute is a function of the distance between each demand point and each planning unit in the attribute space. The shorter the distances between the demand points and the planning units; the better the prioritisation is at representing the variation in the spatial attribute. 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), or are undesirable (eg. conditions known to be outside the physiological tolerance of a species). By placing demand points in desirable regions of an attribute space, the decision maker can ensure that prioritisations secure desirable values of a spatial attribute.

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 be representative in terms of the species' intra-specific variation. To describe this intra-specific variation, the decision maker obtained data on the environmental conditions (rainfall (ml) and tem-

perature (°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, see (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 popuations A and C, they will effectively "double-up" on the same genetic variation, and in turn their waste resources. Instead, a representative sample of the intra-specific variation could be preserved by securing populations A, B, and D.

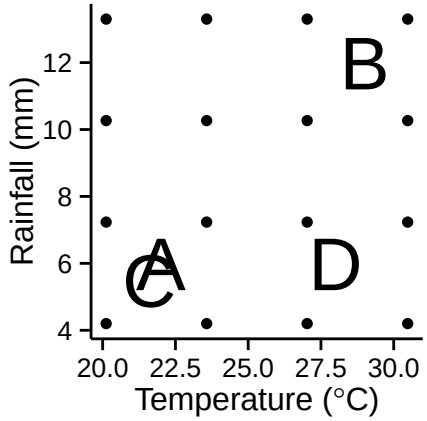


Figure 1: Example of an attribute space. This environmental attribute space has dimensions relating to annual temperature (°C)and rainfall (ml) values. Letters denote the environmental conditions associated with the geographic locations where four hypothetical populations are found. Points represent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

The problems used in the `rapr` R package are based on a combination of the Marxan reserve selection problem and uncapacitated facility location problems (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 boundary length between each planning unit $j \in J$ and $k \in J$ be b_{jk}. In any real-world problem, some planning units will have edges that are not associated with any neighbouring planning units (eg. edges along coastlines), these cases will be denoted using b_{jj} for $j \in J$.

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 $i \in I$ for each feature $f \in F$ and each attribute space $s \in S$. 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 $s \in S$. The decision maker will 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 (Faith *et al.* 1987).

Targets are used to ensure that prioritisations are sufficient in terms of their adequacy and representativeness. Let $\hat{T}_f$ denote the expected amount of area that needs to be preserved for each feature $f \in F$. Let $\bar{T}_{fs}$ denote the representativeness targets for feature $f \in F$ and attribute space $a \in A$. For convenience, these targets are expressed as proportions in the R package between the values for the worst possible prioritisation and the prioritisation that includes all the planning units.

These formulations are based on the unreliable (Fernández and Landete 2015) and reliable uncapacitated facility location (Cui *et al.* 2010) the problems. The key difference between these formulations is that the reliable formulation explicitly considers the probability that the planning units are occupied when determining the level of representation that a prioritisation secures, and the unreliable formulation does not.

Unreliable formulation

The control variables in the unreliable formulation are the X, BLM, $\hat{t}_s$, and $\bar{t}_{sa}$ variables.

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

$$\hat{T}_s = \text{amount target for feature } f \tag{1b}$$

$$\bar{T}_{sa} = \text{representation target for feature } f \text{ in attribute space } a \tag{1c}$$

$$BLM = \text{boundary length modifier} \tag{1d}$$

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

$$\text{(URAP)} \quad \text{Min} \sum_{j=0}^{J-1} C_j + BLM \times \sum_{j=0}^{J-1} \sum_{k=j}^{J-1} X_j \left(1 - X_k\right) b_{jk} + BLM \times \sum_{j=0}^{J-1} x_j b_{jj} \tag{2a}$$

$$\text{s.t.} \sum_{j=0}^{J-1} A_j q_{jf} \geq \bar{T}_f \qquad \forall 0 \leq f \leq F-1 \tag{2b}$$

$$\sum_{i=0}^{I-1} \sum_{j=0}^{J-1} \lambda_{fsi} Y_{fsij} \leq \hat{T}_{fs} \qquad \forall 0 \leq f \leq F-1, \tag{2c}$$
$$0 \leq s \leq S-1$$

$$\sum_{j=0}^{J-1} Y_{fsij} = 1 \qquad \forall 0 \leq f \leq F-1, \tag{2d}$$
$$0 \leq s \leq S-1,$$
$$0 \leq i \leq I-1$$

$$Y_{fsij} \leq X_j \qquad \forall 0 \leq f \leq F-1, \tag{2e}$$
$$0 \leq s \leq S-1,$$
$$0 \leq i \leq I-1,$$
$$0 \leq j \leq J-1$$

$$X_j, Y_{fsij} \in 0, 1 \qquad \forall 0 \leq f \leq F-1, \tag{2f}$$
$$0 \leq s \leq S-1,$$
$$0 \leq i \leq I-1$$

The objective function (2a) determines the utility of a given prioritisation: a combination of the total cost of a prioritisation and how fragmented it is. Constraints (2b–2c) ensure that all the amount-based and space-based targets are met. Constraints (2d) ensure that only one planning unit is assigned to each demand point. Constraints (2e) ensure that demand points are only assigned to selected planning units. Constraints (2f) 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 can 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 achevied by siting back-up planning units near 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 demand points at multiple backup 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 the representation when both 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 meangingless units 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 $f \in F$, $s \in S$ (3).

$$d_{fsiJ} = M \times \max_{0 \le i \le I-1, 0 \le j \le J-1} d_{fsij} \qquad \forall 0 \le f \le F-1, \quad 0 \le s \le S-1 \tag{3}$$

Moderately-sized conservation planning problems often include several thousand planning units. It would simply not be feasible to solve this problem when considering all possible failure scenarios. As a consequence, the R variable can be any $1 \le R \le J-1$. 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 $R = 1$.

The control variables in the reliable formulation are the X, BLM, $\hat{t}_s$, $\bar{t}_{sa}$, R, and M variables.

(1a–1d)

$$R = \text{number of failure levels} \tag{4a}$$

$$M = \text{failure multiplier} \tag{4b}$$

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

(RRAP) Min (2a)

s.t. (2b)

$$\sum_{i=0}^{I-1}\sum_{j=0}^{J}\sum_{r=0}^{R}\lambda_{fsijr}P_{fsijr}Y_{fsijr} \leq \hat{T}_{fs} \qquad \forall 0 \leq f \leq F-1, \quad 0 \leq s \leq S-1 \qquad (5a)$$

$$\sum_{j=0}^{J-1} Y_{fsijr} = 1 \qquad \forall 0 \leq f \leq F-1, \quad 0 \leq s \leq S-1, \quad 0 \leq i \leq I-1, \quad 0 \leq r \leq R \qquad (5b)$$

$$\sum_{r=0}^{R} Y_{fsijr} = 1 \qquad \forall 0 \leq f \leq F-1, \quad 0 \leq s \leq S-1, \quad 0 \leq i \leq I-1, \quad 0 \leq j \leq J \qquad (5c)$$

$$\sum_{r=0}^{R-1} Y_{fsijr} \leq X_j \qquad \forall 0 \leq f \leq F-1, \quad 0 \leq s \leq S-1, \quad 0 \leq i \leq I-1, \quad 0 \leq j \leq J-1 \qquad (5d)$$

$$Y_{fsiJR} = 1 \qquad \forall 0 \leq f \leq F-1, \quad 0 \leq s \leq S-1, \quad 0 \leq i \leq I-1 \qquad (5e)$$

$$P_{fsij0} = q_{fj} \qquad \forall 0 \leq f \leq F-1, \quad 0 \leq s \leq S-1, \quad 0 \leq i \leq I-1, \quad 0 \leq j \leq J \qquad (5f)$$

$$P_{fsijr} = (1-)\sum_{k=0}^{J-1}\frac{1-q_k}{q_k}P_{f,s,i,k,r-1}Y_{f,s,i,k,r-1} \qquad \forall 0 \leq f \leq F-1, \quad 0 \leq s \leq S-1, \quad 0 \leq i \leq I-1, \quad 0 \leq j \leq J, \quad 1 \leq r \leq R \qquad (5g)$$

$$X_j, Y_{fsijr} \in 0, 1 \qquad \forall 0 \leq f \leq F-1, \quad 0 \leq s \leq S-1, \quad 0 \leq i \leq I-1, \quad 0 \leq j \leq J, \quad 0 \leq r \leq R \qquad (5h)$$

The objective function for the reliable formulation is the same as for the unreliable formation (2a). Similar to the unreliable formulation, constraints (2b) and (5a) ensure that the amount-based and space-based targets are met. Constraint (5b–5c) ensure that each planning unit is only assigned to one backup r-level for $i \in I$. Constraints (5d) ensure that only selected planning units are assigned to demand points $i \in I$. Constraints (5e) ensure that the imaginary planning unit is always assigned to the highest backup r-level. Constraints (5f–5g) determine the probability that planing unit j will be used to sample demand point $i \in I$ for $s \in S$ and $f \in F$ (Cui *et al.* 2010). Constraints (5h) 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 (2a) can be linearised using methods described by Beyer et al. (2015). Second, the expression $P_{fsijr} Y_{jsijr}$ in (4a) 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 commerical exact algorithm solvers (eg. IBM CPLEX; Gurobi).

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. Note that academics can easily 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 the in `rapr` R package

Class Name	Description	Slots
ManualOpts	place-holder class for manually specified solutions	NumberSolutions
GurobiOpts	parameters for solving optimisation problems using Gurobi	Threads, MIPGap, Presolve, TimeLimit, NumberSolutions
RapUnreliableOpts	parameters for the unreliable problem formulation	BLM

Class Name	Description	Slots
RapReliableOpts	parameters for the reliable problem formulation	failure.multiplier, max.r.level
SimplePoints	stores coordinates in an n-dimensional space	coords
DemandPoints	demand points coordinates and weights for a speciesin an attribute space	points, weights
AttributeSpace	planning unit coordinates and demand points data for an attribute space	pu, demand.points, distance.metric
RapData	planning unit, species, and attribute space data	pu, species, targets, pu.species.probabilities, attribute.spaces, boundary, polygons, .cache
RapUnsolved	data and parameters needed to generate prioritisationsusing RAP	opts, data
RapResults	prioritisations and summary statistics	summary, selections, amount.held, space.held, logging.file, .cache
RapSolved	data, parameters, and prioritisations using them	opts, data, results

Package tutorial

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.

```
# 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.

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

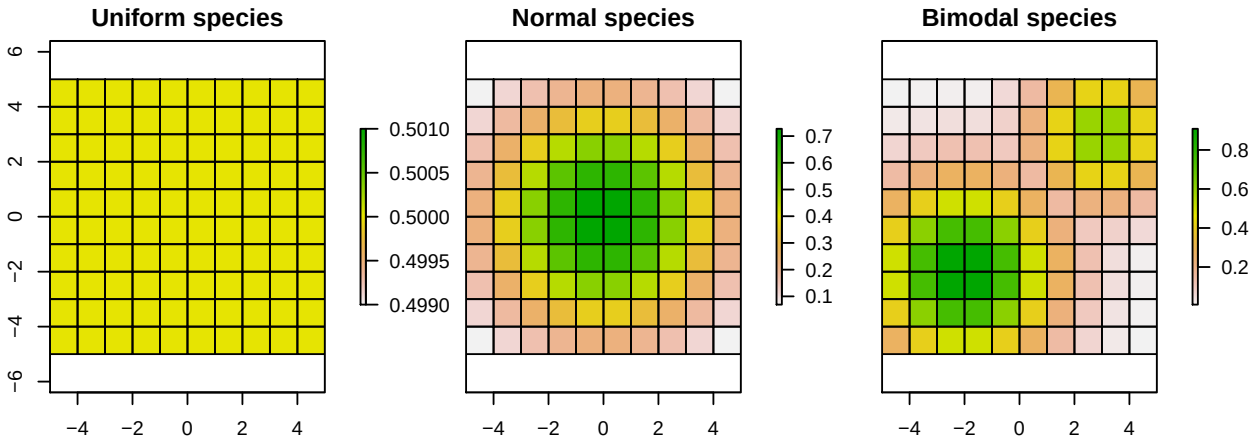


Figure 2: Distribution of three simulated species. Each square represents a planning unit. The color of each square denotes the probability that individuals from each species occupy it.

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 recomended to use different demand points for different species in real-world conservation planning exercises.**

```
# generate coordinates for pus/demand points
pu_coords <- rgeos::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.

```
## create RapUnreliableOpts object
# this stores parameters for the unreliable formulation problem (eg. 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
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 store 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 store all the input data and parameters needed to generate prioritisations
sim_ru <- RapUnsolved(sim_ro, sim_rd)
```

Single-species prioritisations

Amount-based targets

Let's keep things simple to start off with. First we will generate a prioritisation for each species using only amount-based targets. Then, we will generate prioritisations for each species using a combination of amount-based and space-based targets. We will then compare the prioritisations to see how they differ to see how including space-based targets can change prioritisations under the simplest of circumstances.

Now 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 containing data for all the species. 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.

```
# 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=0, solve=FALSE)
```

```
# 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%
```

```
## 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 OPTIMAL    20   20             20                220
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1              42               168               10
##   Connectivity_In_Fraction
## 1                0.1909091
```

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

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

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

```
##   uniform (Space 1)
## 1         0.4859379
```

Now that we have generated a prioritisation, we will see what it looks like. We can use the `plot` method to visualise the spatial distribution of the planning units.

```
# plot the first (and only) solution in the prioritisation
plot(sim_rs_s1_amount, 1)
```

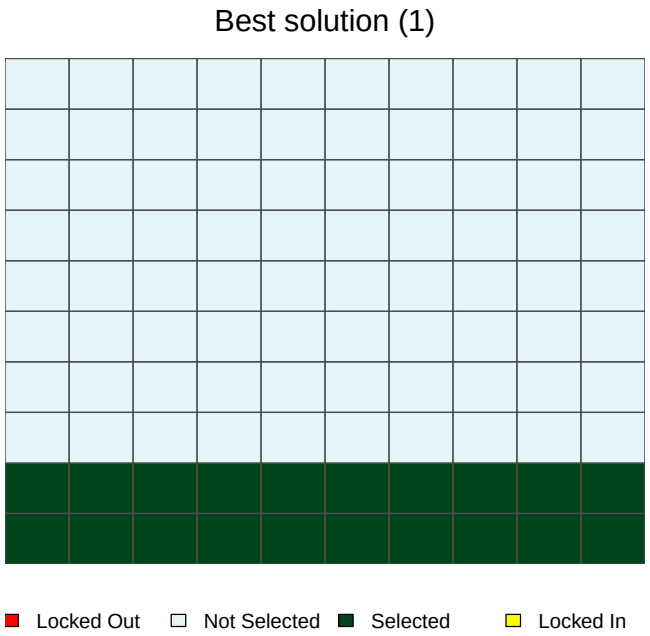


Figure 3: A prioritisation for the uniformly distributed species generated using amount-based targets (20%). Sqaures represent planning units. Planning units with a dark green fill are selected for prioritisation, and those with a pale green color are not.

We can use the `spp.plot` method to see how the prioritisation overlaps with the uniform species' distribution. The fill of the planning units indicates the probability that they are occupied by the species. Since all planning units have equal probabilities for this species, all planning units have the same fill. Planning units with a green border are included for prioritisation.

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

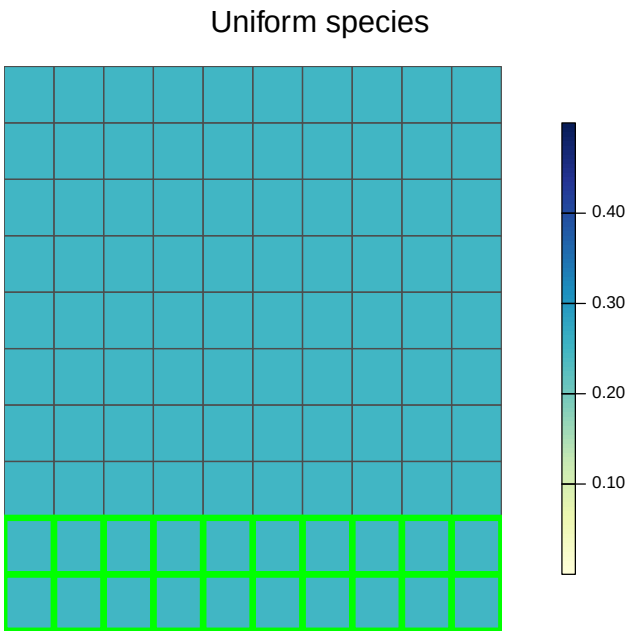


Figure 4: A prioritisation for the uniformly distributed species generated using amount-based targets (20%). Sqaures represent planning units. Planning units with a green border are selected for prioritisation, and their color denotes the probability they are inhabited by the species.

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-targets. We will use a similar process to what we used for the uniformly distributed species, but for brevity, we will generate solutions immediately after updating the object.

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

```
# 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=0, 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: 0.00s
## 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, 0.00 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 0.02 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%
```

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

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

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

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

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

```
##   normal (Space 1)
## 1        0.7456508
```

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

```
# plot the prioritisation
plot(sim_rs_s2_amount, 1)
```

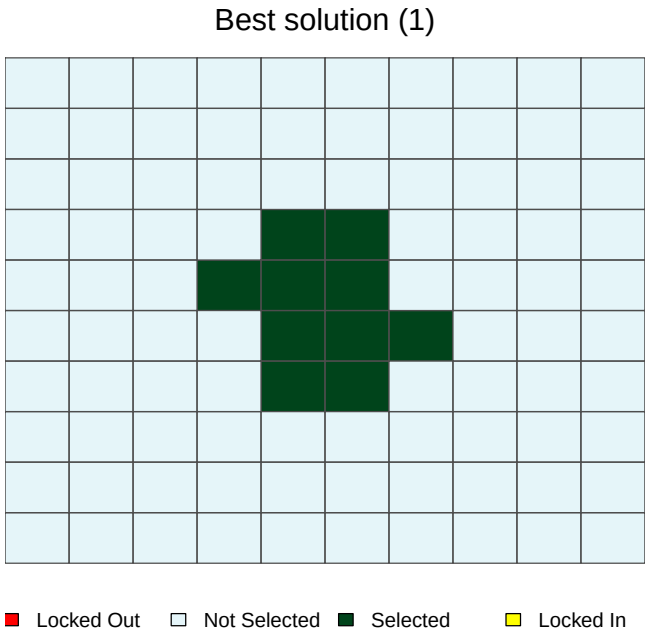


Figure 5: A prioritisation for the normally distributed species generated using amount-based targets (20%). See Figure 3 caption for conventions.

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

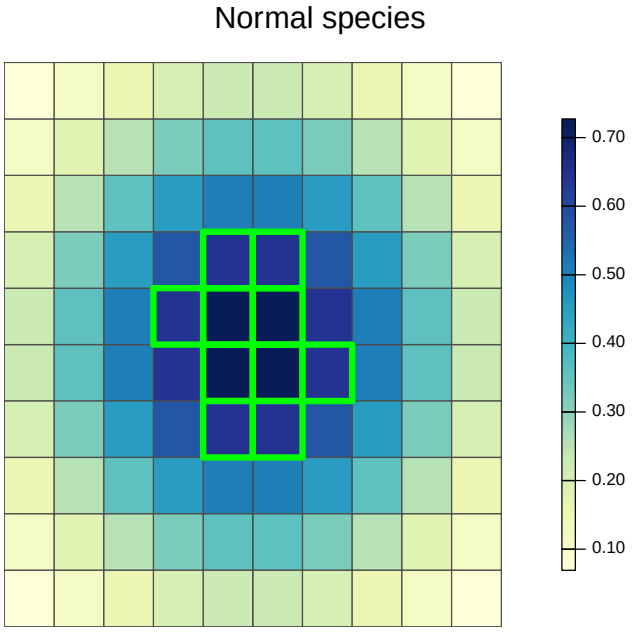


Figure 6: A prioritisation for the normally distributed species generated using amount-based targets (20%). See Figure 4 caption for conventions.

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, and visualise it.

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

```
# 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=0)
```

```
## 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
plot(sim_rs_s3_amount, 1)
```

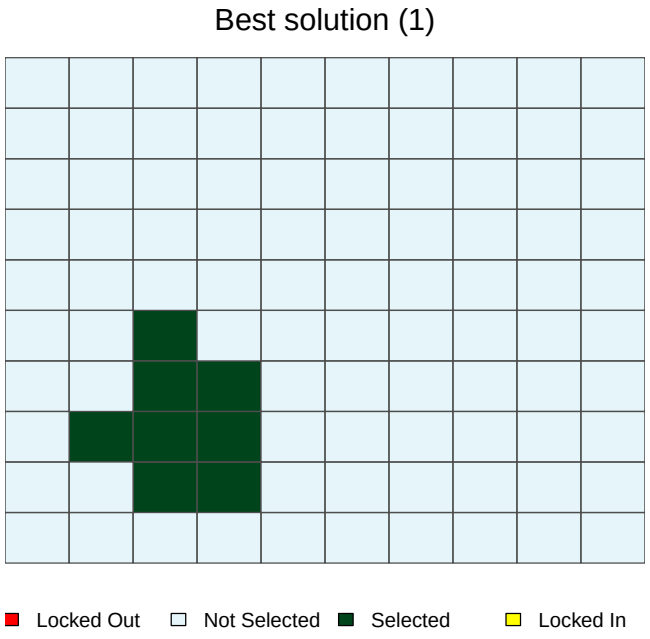


Figure 7: A prioritisation for the bimodally distributed species generated using amount-based targets (20%). See Figure 3 caption for conventions.

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

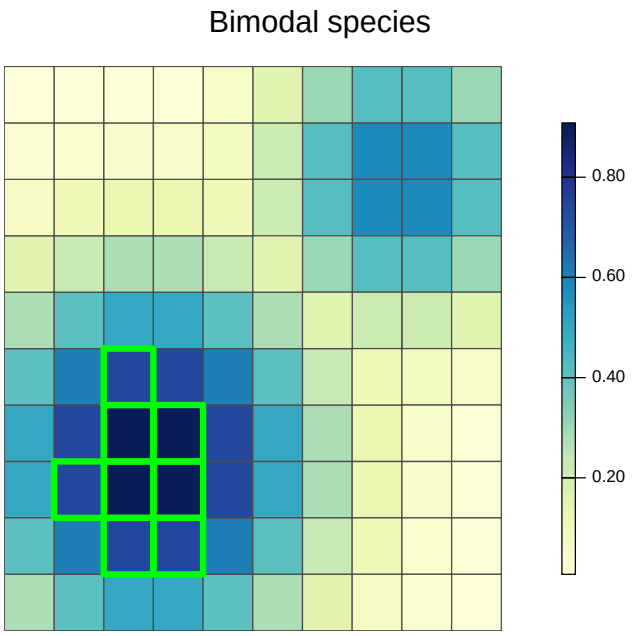


Figure 8: A prioritisation for the bimodally distributed species generated using amount-based targets (20%). See Figure 4 caption for conventions.

```
# 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
```

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

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

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

```
##   bimodal (Space 1)
## 1         0.6514575
```

The amount-based prioritisation for the bimodally distributed species only selects plannig 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 fails 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)
```

```
## Optimize a model with 10102 rows, 10100 columns and 40100 nonzeros
## Coefficient statistics:
##   Matrix range    [1e-05, 6e+00]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 5e+01]
## Found heuristic solution: objective 83
## Presolve time: 0.98s
## Presolved: 10102 rows, 10100 columns, 40100 nonzeros
## Variable types: 0 continuous, 10100 integer (10100 binary)
## Presolved: 10102 rows, 10100 columns, 40100 nonzeros
##
## Presolve removed 10001 rows and 6276 columns
##
## Root relaxation: objective 2.000000e+01, 1237 iterations, 0.33 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%     -    1s
##
## Explored 0 nodes (1239 simplex iterations) in 1.39 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     8    8              8                220
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1               9               197               14
##   Connectivity_In_Fraction
## 1               0.04090909
```

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

```
##   uniform
## 1    0.08
```

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

```
##   uniform (Space 1)
## 1         0.4989372
```

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.

```
# plot the prioritisation
plot(sim_rs_s1_space, 1)
```

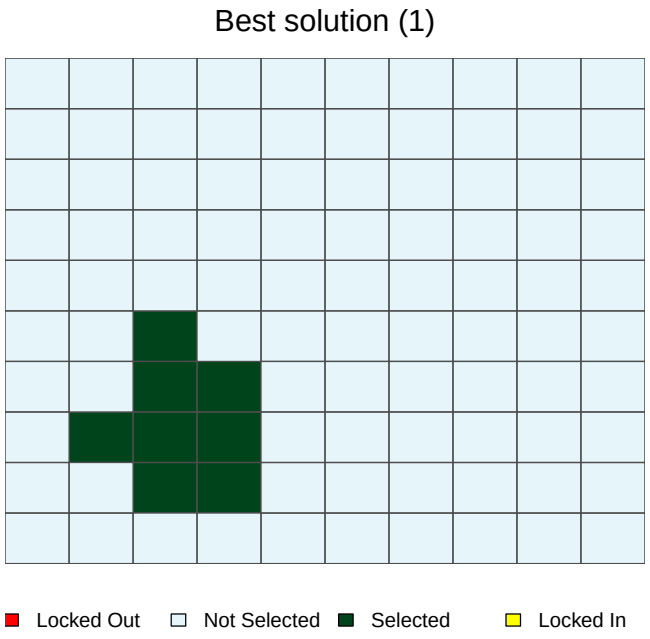


Figure 9: A prioritisation for the uniformly distributed species generated using amount-based targets (20%) and space-based targets (85%). See Figure 3 caption for conventions.

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

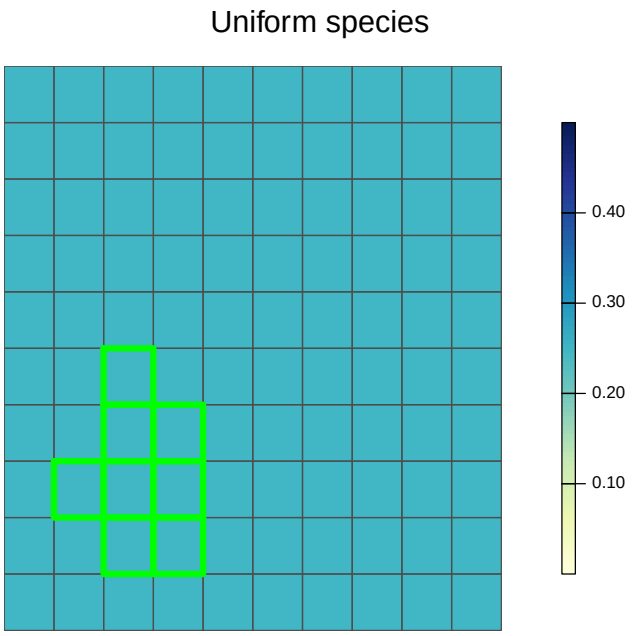


Figure 10: A prioritisation for the uniformly distributed species generated using amount-based targets (20%) and space-based targets (85%). See Figure 4 caption for conventions.

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

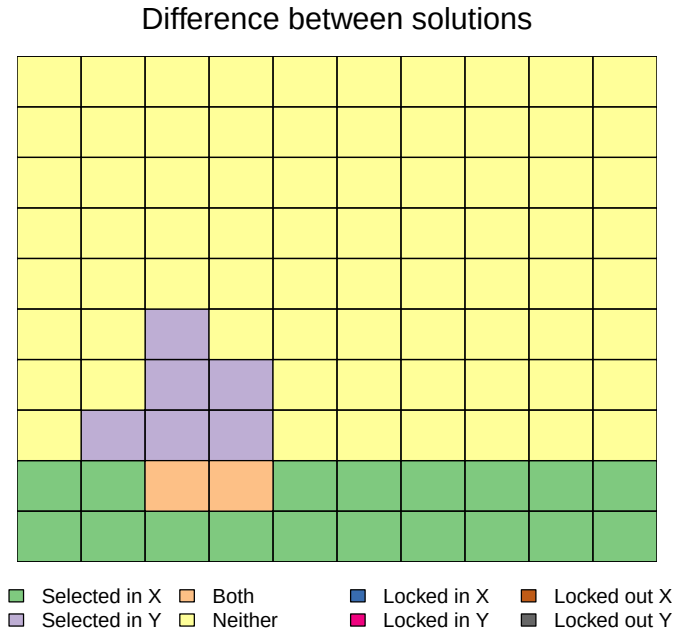


Figure 11: Difference between two prioritisations for the uniformly distributed species. Prioritisation X was generated using just amount-based targets (20%), and prioritisation Y was generated using an additional space-based target (85%).

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.

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

```
## Optimize a model with 10102 rows, 10100 columns and 40100 nonzeros
## Coefficient statistics:
##   Matrix range    [1e-05, 5e+00]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 3e+01]
## Found heuristic solution: objective 81
## Presolve time: 0.96s
## Presolved: 10102 rows, 10100 columns, 40100 nonzeros
## Variable types: 0 continuous, 10100 integer (10100 binary)
## Presolved: 10102 rows, 10100 columns, 40100 nonzeros
##
## Presolve removed 10001 rows and 6276 columns
```

```
##
## Root relaxation: objective 1.297069e+01, 4408 iterations, 0.93 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0   12.97069    0  316   81.00000   12.97069  84.0%     -    2s
## H    0     0                      14.0000000   12.97069  7.35%     -    2s
##      0     0   12.97069    0  360   14.00000   12.97069  7.35%     -    3s
##      0     0     cutoff    0        14.00000   13.00002  7.14%     -    5s
##
## Explored 0 nodes (5159 simplex iterations) in 5.96 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 1.400000000000e+01, best bound 1.400000000000e+01, gap 0.0%
```

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

```
##   Run_Number Status Score Cost Planning_Units Connectivity_Total
## 1          1 MANUAL    14   14             14                220
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1               4               168               48
##   Connectivity_In_Fraction
## 1               0.01818182
```

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

```
##      normal
## 1 0.2154063
```

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

```
##   normal (Space 1)
## 1        0.8536759
```

```
# plot the prioritisation
plot(sim_rs_s2_space, 1)
```

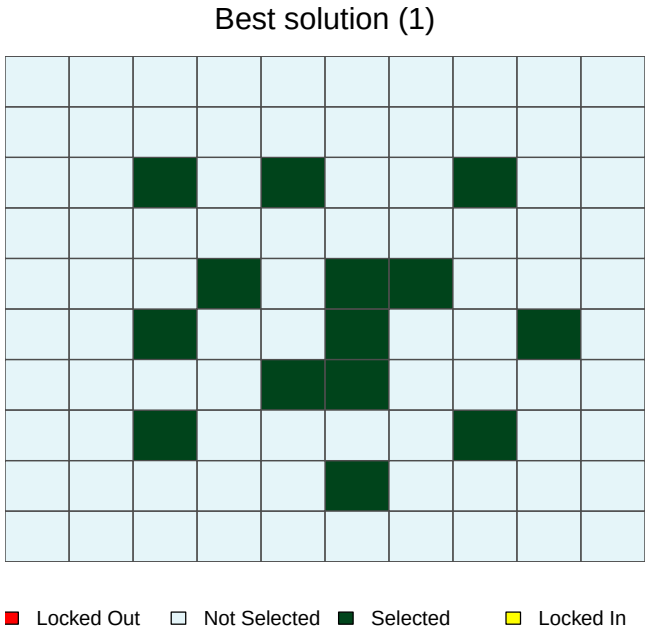


Figure 12: A prioritisation for the normally distributed species generated using amount-based targets (20%) and space-based targets (85%). See Figure 3 caption for conventions.

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

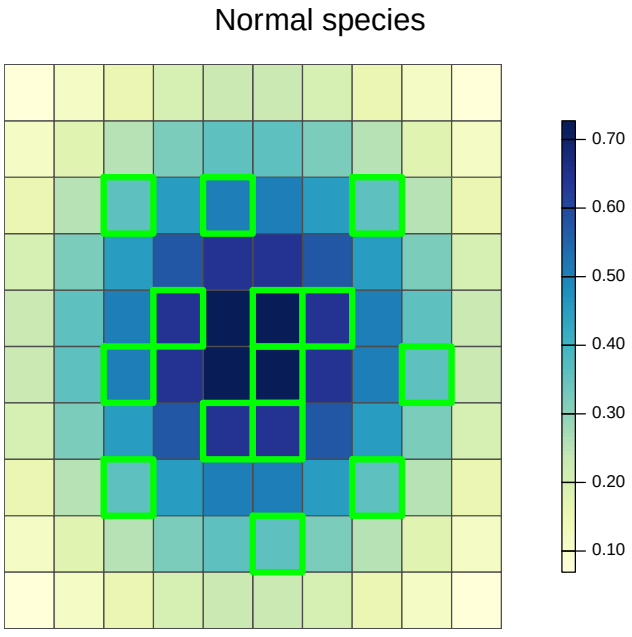


Figure 13: A prioritisation for the normally distributed species generated using amount-based targets (20%) and space-based targets (85%). See Figure 4 caption for conventions.

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

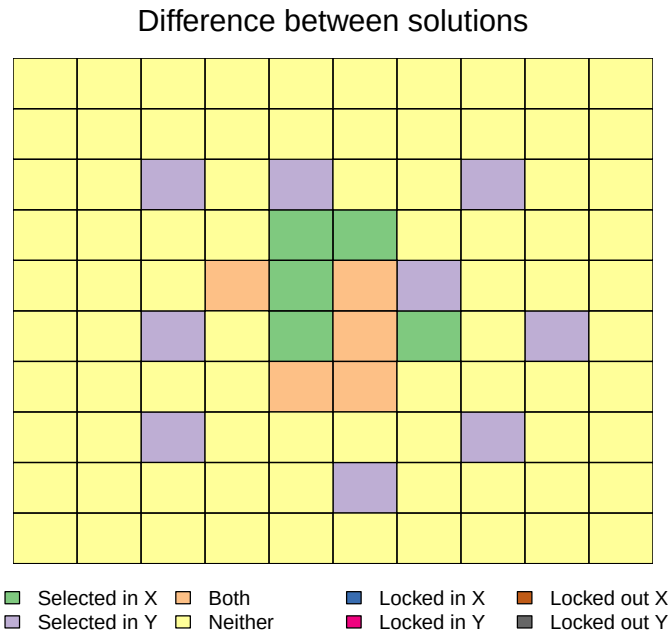


Figure 14: Difference between two prioritisations for the normally distributed species. See Figure 11 caption for conventions.

We can see by using both amount-based and space-based targets we can obtain a priortisation 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)
```

```
## Optimize a model with 10102 rows, 10100 columns and 40100 nonzeros
## Coefficient statistics:
##   Matrix range    [1e-05, 9e+00]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 4e+01]
## Found heuristic solution: objective 79
## Presolve time: 0.95s
## Presolved: 10102 rows, 10100 columns, 40100 nonzeros
## Variable types: 0 continuous, 10100 integer (10100 binary)
```

```
## Presolved: 10102 rows, 10100 columns, 40100 nonzeros
##
## Presolve removed 10001 rows and 6276 columns
##
## Root relaxation: objective 9.615676e+00, 6227 iterations, 1.40 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0    9.61568    0   35   79.00000    9.61568  87.8%     -    2s
## H    0     0                      10.0000000    9.61568  3.84%     -    2s
##
## Explored 0 nodes (6486 simplex iterations) in 2.63 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%
```

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

```
##   Run_Number Status Score Cost Planning_Units Connectivity_Total
## 1          1 MANUAL    10   10             10                220
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1               3               183               34
##   Connectivity_In_Fraction
## 1               0.01363636
```

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

```
##     bimodal
## 1 0.2107345
```

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

```
##   bimodal (Space 1)
## 1         0.8514121
```

```
# plot the prioritisation
plot(sim_rs_s3_space, 1)
```

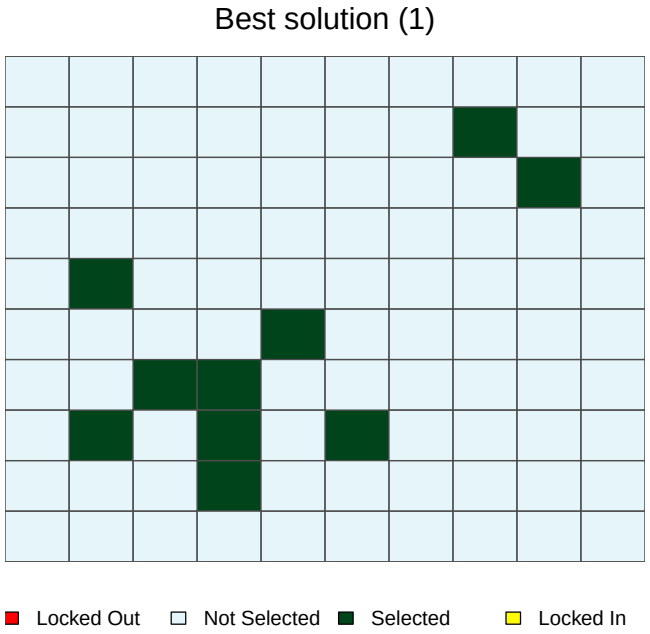


Figure 15: A prioritisation for the bimodally distributed species generated using amount-based targets (20%) and space-based targets (85%). See Figure 3 caption for conventions.

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

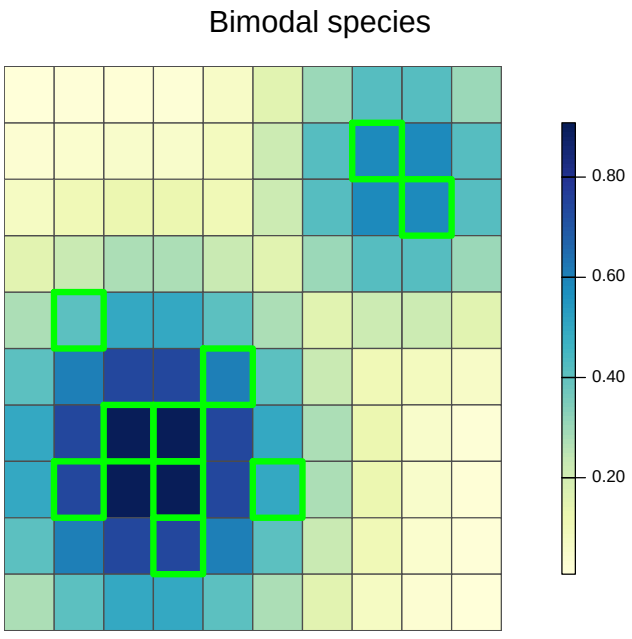


Figure 16: A prioritisation for the normally distributed species generated using amount-based targets (20%) and space-based targets (85%). See Figure 4 caption for conventions.

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

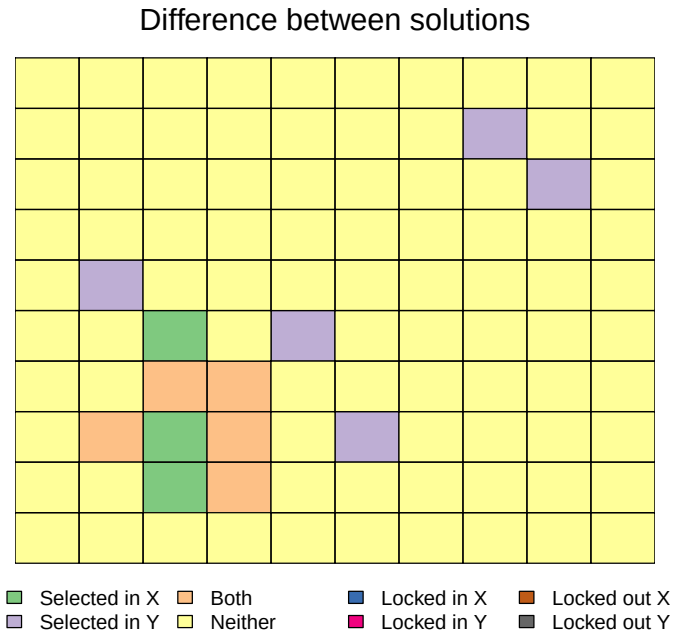


Figure 17: Difference between two prioritisations for the bimodally distributed species. See Figure 11 caption for conventions.

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 the 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 that the use of space-based targets can improve prioritisations. However, all these prioritisations were generated for a single species at a time. It is possible that by prioritisations generated using multiple species may do a better job at preserving the intra-specific variation for individuals species. 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 for 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 uses space-based targets to also preserve 85% of their geographic distribution. After doing this we will compare the two prioritisations.

```r
# make prioritisations
sim_mrs_amount <- update(sim_ru, amount.target=c(0.2,0.2,0.2), space.target=c(0,0,0))
```

```
## Optimize a model with 3 rows, 100 columns and 300 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, 1e+01]
## Found heuristic solution: objective 27
## Presolve removed 0 rows and 45 columns
## Presolve time: 0.00s
## Presolved: 3 rows, 55 columns, 165 nonzeros
## Variable types: 0 continuous, 55 integer (10 binary)
## Presolved: 3 rows, 55 columns, 165 nonzeros
##
##
## Root relaxation: objective 2.000000e+01, 1 iterations, 0.00 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%     -    0s
##
## Explored 0 nodes (1 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%
```

```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))
```

```
## Optimize a model with 30306 rows, 30100 columns and 120300 nonzeros
## Coefficient statistics:
##   Matrix range    [1e-05, 9e+00]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 5e+01]
## Found heuristic solution: objective 100
## Presolve removed 0 rows and 0 columns (presolve time = 5s) ...
## Presolve time: 6.16s
## Presolved: 30306 rows, 30100 columns, 120300 nonzeros
## Variable types: 0 continuous, 30100 integer (30100 binary)
## Presolved: 30306 rows, 30100 columns, 120300 nonzeros
##
## Presolve removed 30003 rows and 18628 columns
```

```
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    0.0000000e+00   6.836079e+01   9.572551e+08      8s
##     4648    2.0000000e+01   0.000000e+00   0.000000e+00     10s
##     4648    2.0000000e+01   0.000000e+00   0.000000e+00     10s
##
## Root relaxation: objective 2.000000e+01, 4648 iterations, 3.38 seconds
## Total elapsed time = 10.22s
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0   20.00000    0   42  100.00000   20.00000  80.0%     -   11s
## H    0     0                      21.0000000   20.00000  4.76%     -   11s
##
## Explored 0 nodes (8779 simplex iterations) in 11.75 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 2.100000000000e+01, best bound 2.000000000000e+01, gap 4.7619%
```

```
# 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              20               158               42
##   Connectivity_In_Fraction
## 1               0.09090909
```

```
summary(sim_mrs_space)
```

```
##   Run_Number Status Score Cost Planning_Units Connectivity_Total
## 1          1 MANUAL    21   21             21                220
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1              11               141               68
##   Connectivity_In_Fraction
## 1                     0.05
```

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

```
##   uniform    normal   bimodal
## 1     0.2 0.2805844 0.3255251
```

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

```
##   uniform    normal   bimodal
## 1    0.21 0.2417393 0.2257705
```

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

```
##   uniform (Space 1) normal (Space 1) bimodal (Space 1)
## 1         0.7840732        0.8333445         0.8320986
```

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

```
##   uniform (Space 1) normal (Space 1) bimodal (Space 1)
## 1           0.86966        0.8777045         0.8826188
```

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

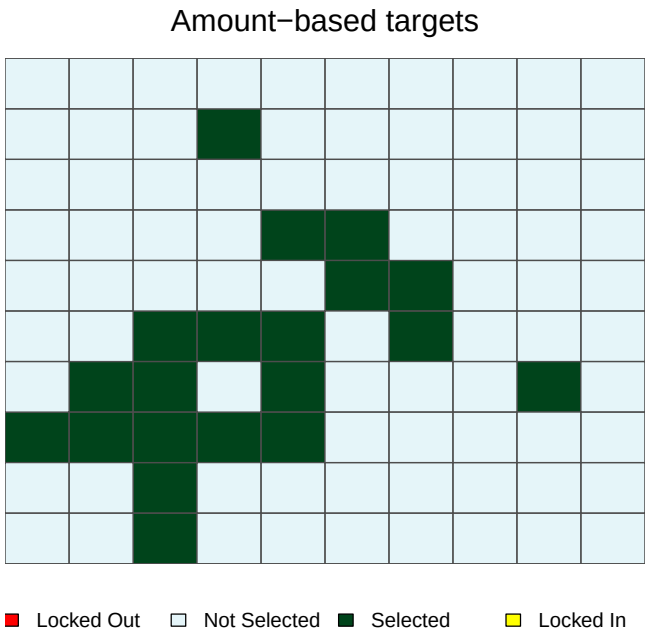


Figure 18: A multi-species prioritisation for the uniformly, normally, and bimodally distributed species generated using just amount-based targets (20%). See Figure 3 caption for conventions.

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

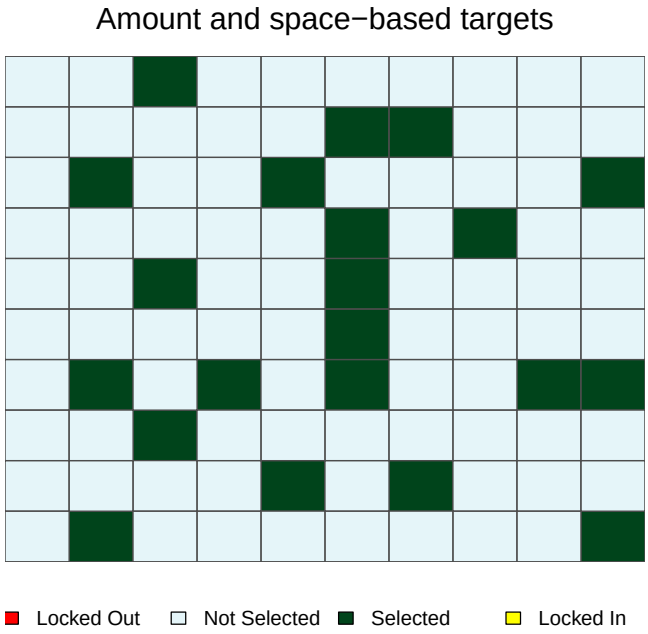


Figure 19: A multi-species prioritisation for the uniformly, normally, and bimodally distributed species generated using amount-based targets (20%) and space-based targets (85%). See Figure 3 caption for conventions.

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

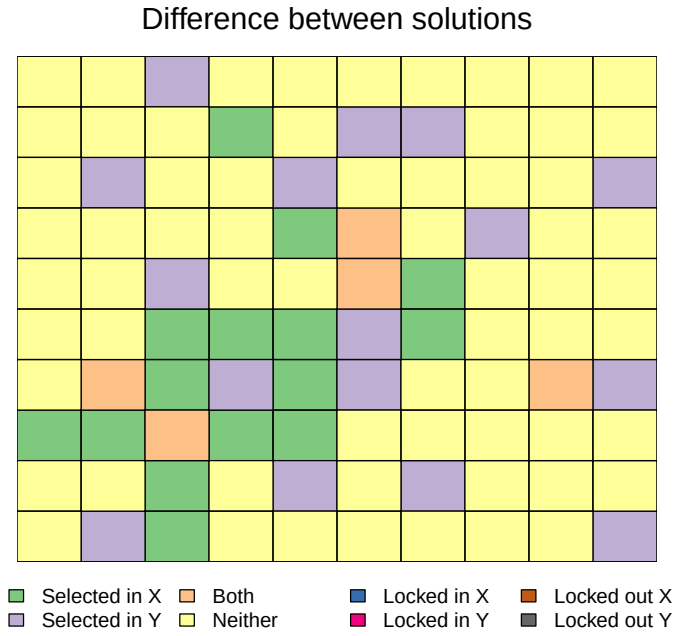


Figure 20: Difference between two multi-species prioritisations. See Figure 11 caption for conventions.

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 prioritistion is comprised of 20 units, and the space-based prioritisation is comprised of 21 units. This result suggests that a adequate and representative prioritisation can be acheived 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 represent regions of an attribute space for a species using planning units only have a small chance of being occupied by the species. 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 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 acheived 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(0.5,0.5,0.5)
    )
)
```

```
## Optimize a model with 14706 rows, 14500 columns and 57744 nonzeros
## Coefficient statistics:
##   Matrix range    [1e-05, 8e+00]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 5e+01]
## Presolve time: 1.96s
## Presolved: 14706 rows, 14500 columns, 57558 nonzeros
## Variable types: 0 continuous, 14500 integer (14500 binary)
## Found heuristic solution: objective 100.0000000
## Found heuristic solution: objective 89.0000000
## Presolved: 14706 rows, 14500 columns, 57558 nonzeros
##
## Presolve removed 14559 rows and 10135 columns
##
## Root relaxation: objective 2.000000e+01, 1965 iterations, 0.58 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%     -    2s
##
## Explored 0 nodes (2636 simplex iterations) in 2.92 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_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               144               69
##   Connectivity_In_Fraction
## 1               0.03181818
```

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

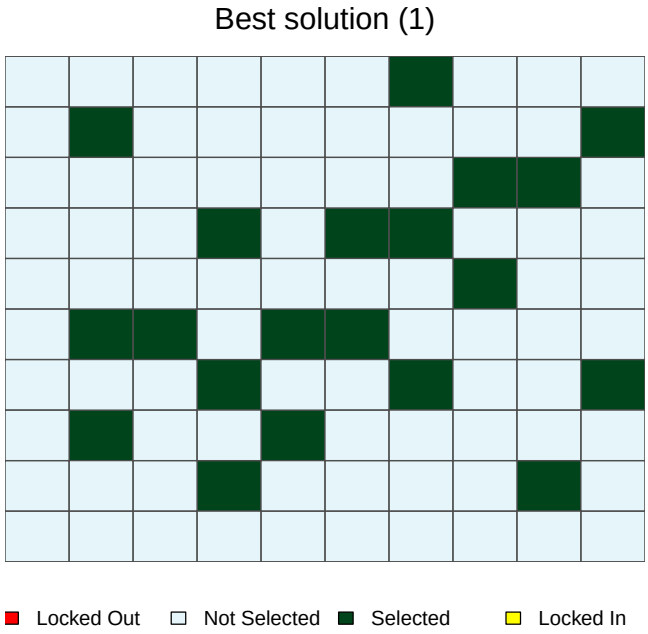


Figure 21: A multi-species prioritisation for the uniformly, normally, and bimodally distributed species generated using amount-based targets (20%) and space-based targets (85%). This prioritisation was generated to be robust against low occupancy probabilities, by preventing planning units with low probabilities from being used to represent demand points. See Figure 3 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')
```

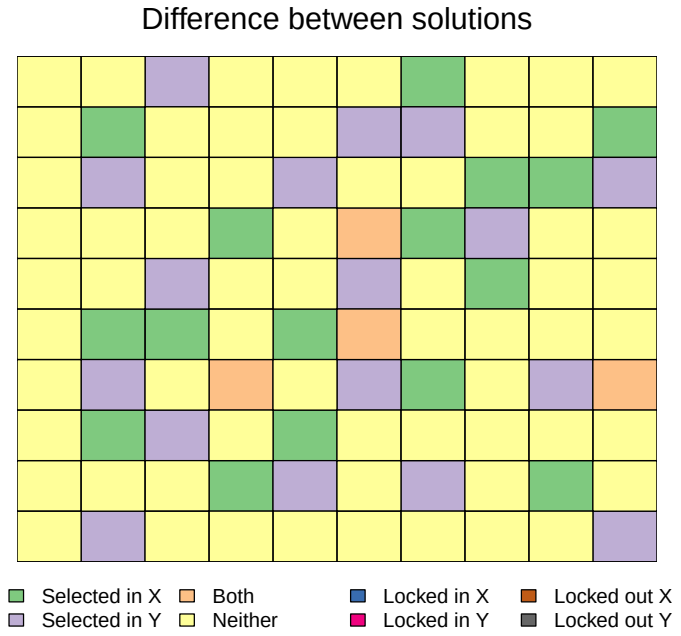


Figure 22: Difference between two multi-species prioritisations. See Figure 11 caption for conventions.

But this method requires setting somewhat arbitrary thresholds. A more objective solution to this issue would be to actually use the probability that species occupy planning units to generate the prioritisations–this is what the reliable formulation does. Now we will generate a prioritisation using 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 <- update(sim_mrs_space, formulation='reliable', max.r.level=1L)
```

```
## Optimize a model with 364206 rows, 181900 columns and 3847800 nonzeros
## Coefficient statistics:
##   Matrix range    [1e-05, 1e+02]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [7e-03, 7e+02]
## Presolve removed 333500 rows and 90800 columns (presolve time = 5s) ...
## Presolve removed 333700 rows and 151600 columns (presolve time = 10s) ...
## Presolve removed 333700 rows and 151600 columns (presolve time = 15s) ...
## Presolve removed 333700 rows and 151600 columns (presolve time = 141s) ...
## Presolve removed 333700 rows and 151600 columns
## Presolve time: 144.01s
## Presolved: 30506 rows, 30300 columns, 139889 nonzeros
## Variable types: 200 continuous, 30100 integer (30100 binary)
## Found heuristic solution: objective 100.0000000
## Presolved: 30506 rows, 30300 columns, 139889 nonzeros
```

```
##
## Presolve removed 30003 rows and 7966 columns
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    0.0000000e+00   6.928672e+01   6.376481e+08    149s
##     1669    2.0000000e+01   0.000000e+00   0.000000e+00    149s
##     1669    2.0000000e+01   0.000000e+00   0.000000e+00    149s
##
## Root relaxation: objective 2.000000e+01, 1669 iterations, 1.75 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0   20.00000    0    2  100.00000   20.00000  80.0%     -  149s
## H    0     0                      20.0000000   20.00000  0.00%     -  149s
##
## Explored 0 nodes (3546 simplex iterations) in 149.65 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
## greater than 1, consider increasing the failure.multiplier
```

```
# 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               9               144               67
##   Connectivity_In_Fraction
## 1               0.04090909
```

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

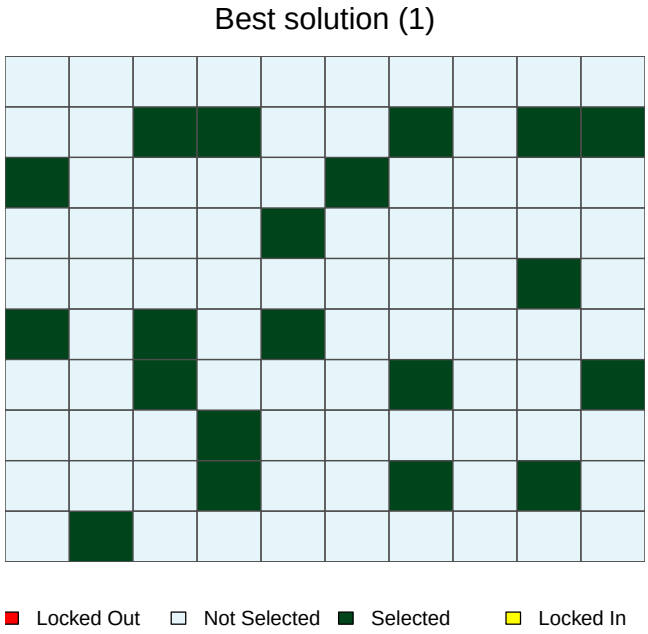


Figure 23: A multi-species prioritisation for the uniformly, normally, and bimodally distributed species generated using amount-based targets (20%) and space-based targets (85%). This prioritisation was generated to be robust against low occupancy probabilities, by explicitly using the probability data when deriving a solution. See Figure 3 caption for conventions.

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

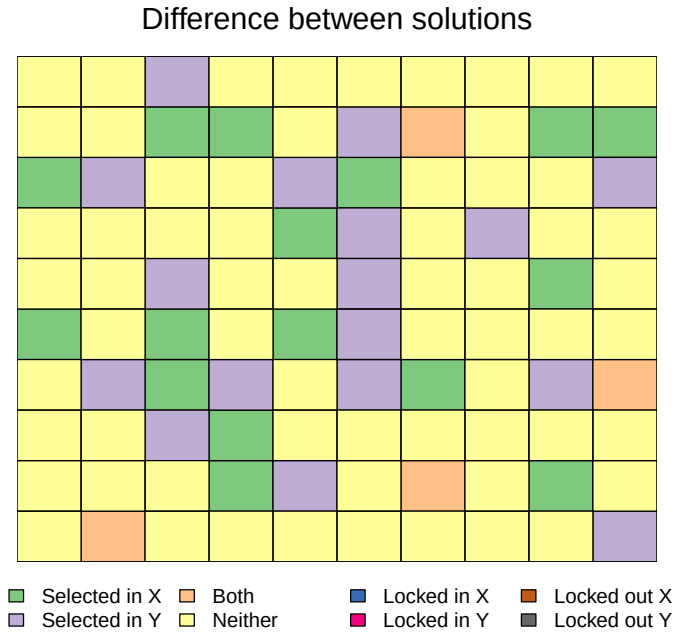


Figure 24: Difference between two multi-species prioritisations. See Figure 11 caption for conventions.

More planning units were selected using the reliable formulation. The prioritisation based on the unreliable formulation had 21 planning units, but the priortisation based on the reliable formlation has 20 planning units. This 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 only 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 neighbors. We can use the `BLM` parameter to penalise fragmented solutions.

Let's generate a new prioritisation that heavily penalises fragmention. 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)
```

```
## Optimize a model with 363 rows, 280 columns and 1020 nonzeros
## Coefficient statistics:
##   Matrix range    [7e-03, 1e+00]
##   Objective range [1e+02, 4e+02]
```

```
##   Bounds range     [1e+00, 1e+00]
##   RHS range        [7e+00, 1e+01]
## Found heuristic solution: objective 2727
## Presolve time: 0.01s
## Presolved: 363 rows, 280 columns, 1020 nonzeros
## Variable types: 0 continuous, 280 integer (280 binary)
## Presolved: 363 rows, 280 columns, 1020 nonzeros
##
##
## Root relaxation: objective 4.644444e+02, 356 iterations, 0.01 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0  464.44444    0  225 2727.00000  464.44444  83.0%     -    0s
## H    0     0                    1220.0000000  464.44444  61.9%     -    0s
##      0     0  525.06329    0  218 1220.00000  525.06329  57.0%     -    0s
##      0     0  560.87213    0  212 1220.00000  560.87213  54.0%     -    0s
##      0     0  608.69464    0  208 1220.00000  608.69464  50.1%     -    0s
##      0     0  609.29259    0  209 1220.00000  609.29259  50.1%     -    0s
## H    0     0                    1120.0000000  609.29259  45.6%     -    0s
##      0     2  610.05238    0  209 1120.00000  610.05238  45.5%     -    0s
## H   13    11                     920.0000000  610.05238  33.7%  15.3    0s
##
## Cutting planes:
##   Gomory: 4
##
## Explored 587 nodes (5436 simplex iterations) in 1.27 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 9.200000000000e+02, best bound 8.760000000000e+02, gap 4.7826%
```

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

```
##   Run_Number Status Score Cost Planning_Units Connectivity_Total
## 1          1 MANUAL  1820   20             20                220
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1              31               171               18
##   Connectivity_In_Fraction
## 1                0.1409091
```

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

```
##   uniform    normal   bimodal
```

```
## 1     0.2 0.2736046 0.4224767
```

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

```
##   uniform (Space 1) normal (Space 1) bimodal (Space 1)
## 1         0.6652267        0.7303832         0.7902917
```

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

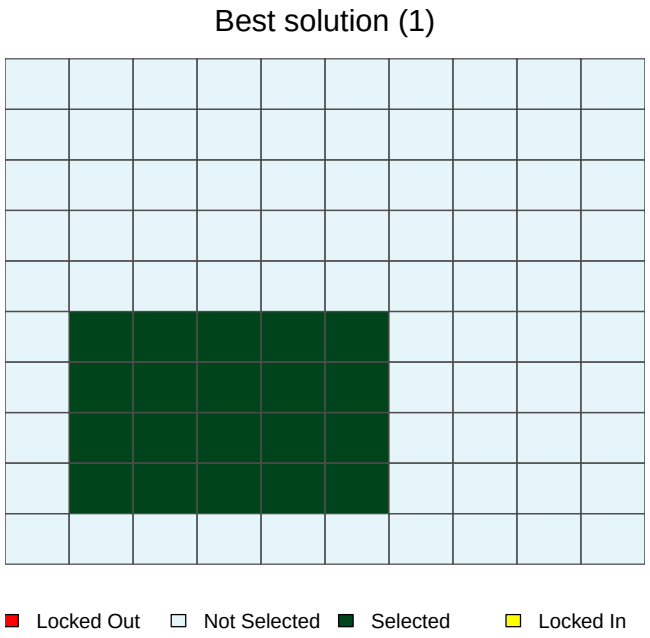


Figure 25: A multi-species prioritisation for the uniformly, normally, and bimodally distributed species generated using only amount-based targets (20%). Additionally, this prioritititisation was generated to be well-connected, by using a high BLM parameter. See Figure 3 caption for conventions.

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

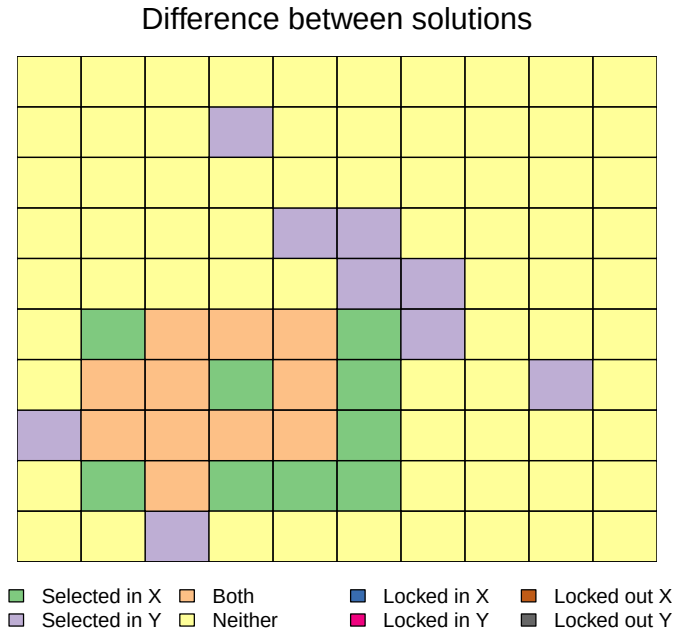


Figure 26: Difference between two multi-species prioritisations. See Figure 11 caption for conventions.

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. To illustrate how using different distance metrics can affect the the optimal 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, the 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.

```
# load RandomFields package and set seed for simulations
library(RandomFields)
set.seed(500)
```

```
# simulate planning units
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)
```

```
## ...
```

```
# 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, 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.

```
# 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)
```

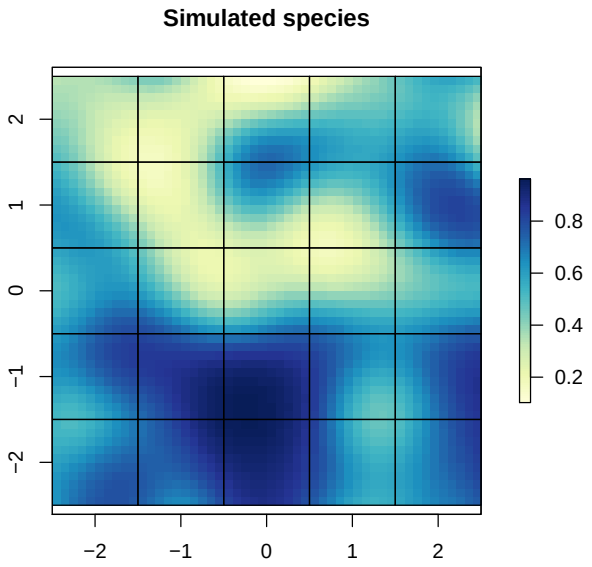


Figure 27: Distribution map of a species simulated using Gaussian prorceses. See Figure 2 caption for conventions.

```
# 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
)
```

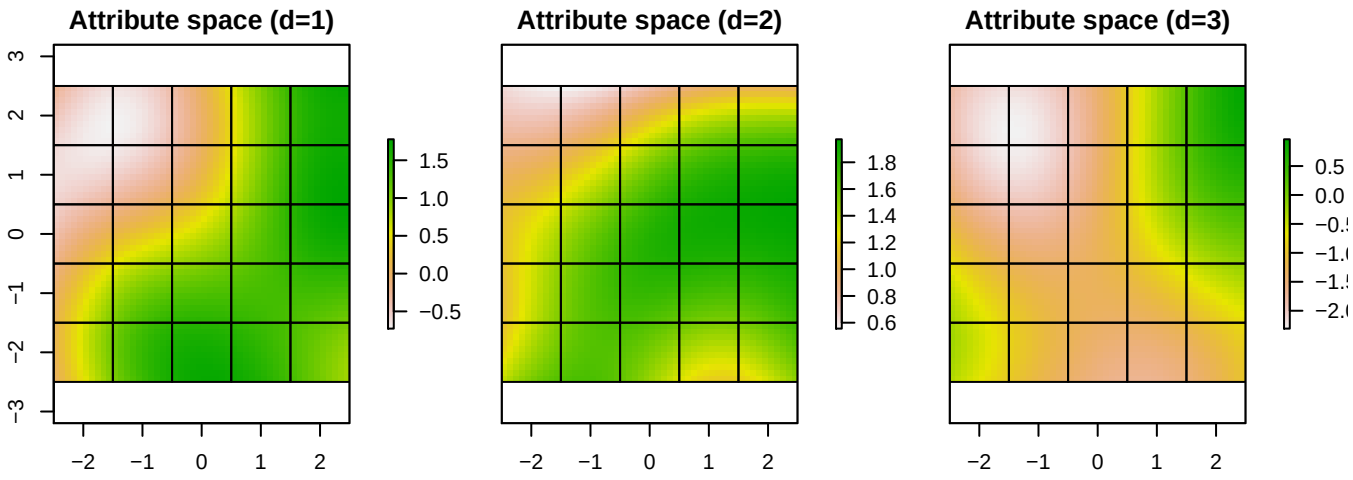


Figure 28: Distribution of spatial variables across the species' geographic range. These variables each represent a dimension of a three-dimensional attribute space.

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+02]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 8e+02]
## Presolve removed 8 rows and 0 columns
## Presolve time: 0.10s
## Presolved: 1294 rows, 1275 columns, 4998 nonzeros
## Variable types: 0 continuous, 1275 integer (1275 binary)
## Found heuristic solution: objective 21.0000000
## Found heuristic solution: objective 19.0000000
## Presolved: 1294 rows, 1275 columns, 4998 nonzeros
##
## Presolve removed 1294 rows and 1275 columns
##
## Root relaxation: objective 4.056382e+00, 795 iterations, 0.06 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0    4.05638    0  102   19.00000    4.05638  78.7%     -    0s
## H    0     0                      17.0000000    4.05638  76.1%     -    0s
## H    0     0                       5.0000000    4.05638  18.9%     -    0s
##
## Explored 0 nodes (795 simplex iterations) in 0.21 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, 3e+05]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 3e+04]
## Found heuristic solution: objective 21
## Presolve removed 2 rows and 1 columns
## Presolve time: 0.06s
## Presolved: 1300 rows, 1274 columns, 3772 nonzeros
## Variable types: 0 continuous, 1274 integer (1274 binary)
## Presolved: 1300 rows, 1274 columns, 3772 nonzeros
##
## Presolve removed 1300 rows and 1274 columns
##
## Root relaxation: objective 3.274119e+00, 189 iterations, 0.03 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0    3.27412    0    1   21.00000    3.27412  84.4%     -    0s
## H    0     0                       4.0000000    3.27412  18.1%     -    0s
##
## Explored 0 nodes (497 simplex iterations) in 0.10 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 4.000000000000e+00, best bound 4.000000000000e+00, gap 0.0%
## Optimize a model with 1302 rows, 1275 columns and 5025 nonzeros
## Coefficient statistics:
##   Matrix range    [3e-01, 5e+02]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 1e+03]
## Presolve removed 8 rows and 0 columns
## Presolve time: 0.10s
## Presolved: 1294 rows, 1275 columns, 4999 nonzeros
## Variable types: 0 continuous, 1275 integer (1275 binary)
## Found heuristic solution: objective 19.0000000
## Found heuristic solution: objective 17.0000000
## Presolved: 1294 rows, 1275 columns, 4999 nonzeros
##
## Presolve removed 1294 rows and 1275 columns
##
## Root relaxation: objective 3.874090e+00, 998 iterations, 0.07 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
```

```
##
##      0     0    3.87409    0   69   17.00000    3.87409  77.2%     -    0s
## H    0     0                       4.0000000    3.87409  3.15%     -    0s
##
## Explored 0 nodes (998 simplex iterations) in 0.20 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 4.000000000000e+00, best bound 4.000000000000e+00, gap 0.0%
## Optimize a model with 1302 rows, 1275 columns and 5025 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, 9e+01]
## Presolve removed 10 rows and 0 columns
## Presolve time: 0.10s
## Presolved: 1292 rows, 1275 columns, 4996 nonzeros
## Variable types: 0 continuous, 1275 integer (1275 binary)
## Found heuristic solution: objective 18.0000000
## Found heuristic solution: objective 17.0000000
## Presolved: 1292 rows, 1275 columns, 4996 nonzeros
##
## Presolve removed 1292 rows and 1275 columns
##
## Root relaxation: objective 3.845913e+00, 727 iterations, 0.05 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0    3.84591    0   69   17.00000    3.84591  77.4%     -    0s
## H    0     0                       4.0000000    3.84591  3.85%     -    0s
##
## Explored 0 nodes (727 simplex iterations) in 0.18 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 4.000000000000e+00, best bound 4.000000000000e+00, gap 0.0%
## Optimize a model with 1302 rows, 1275 columns and 5025 nonzeros
## Coefficient statistics:
##   Matrix range    [3e-01, 2e+02]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 8e+02]
## Presolve removed 13 rows and 0 columns
## Presolve time: 0.10s
## Presolved: 1289 rows, 1275 columns, 4993 nonzeros
## Variable types: 0 continuous, 1275 integer (1275 binary)
```

```
## Found heuristic solution: objective 20.0000000
## Found heuristic solution: objective 15.0000000
## Presolved: 1289 rows, 1275 columns, 4993 nonzeros
##
## Presolve removed 1289 rows and 1275 columns
##
## Root relaxation: objective 4.197266e+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    4.19727    0   79   15.00000    4.19727  72.0%     -    0s
## H    0     0                       5.0000000    4.19727  16.1%     -    0s
##
## Explored 0 nodes (571 simplex iterations) in 0.17 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, 2e+02]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 8e+02]
## Presolve removed 3 rows and 0 columns
## Presolve time: 0.08s
## Presolved: 1299 rows, 1275 columns, 4978 nonzeros
## Variable types: 0 continuous, 1275 integer (1275 binary)
## Found heuristic solution: objective 21.0000000
## Found heuristic solution: objective 19.0000000
## Presolved: 1299 rows, 1275 columns, 4978 nonzeros
##
## Presolve removed 1299 rows and 1275 columns
##
## Root relaxation: objective 4.347763e+00, 663 iterations, 0.05 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0    4.34776    0   86   19.00000    4.34776  77.1%     -    0s
## H    0     0                       5.0000000    4.34776  13.0%     -    0s
##
## Explored 0 nodes (663 simplex iterations) in 0.17 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, 3e+03]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 3e+03]
## Found heuristic solution: objective 17
## Presolve time: 0.08s
## Presolved: 1302 rows, 1275 columns, 5003 nonzeros
## Variable types: 0 continuous, 1275 integer (1275 binary)
## Presolve removed 4 rows and 0 columns
## Presolved: 1298 rows, 1275 columns, 4991 nonzeros
##
## Presolve removed 1269 rows and 1063 columns
##
## Root relaxation: objective 3.622614e+00, 812 iterations, 0.06 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0    3.62261    0   18   17.00000    3.62261  78.7%     -    0s
## H    0     0                       5.0000000    3.62261  27.5%     -    0s
##      0     0    3.78199    0   89    5.00000    3.78199  24.4%     -    0s
##      0     0    3.78199    0   17    5.00000    3.78199  24.4%     -    0s
##      0     0    3.78199    0   88    5.00000    3.78199  24.4%     -    0s
## H    0     0                       4.0000000    3.78199  5.45%     -    0s
##
## Cutting planes:
##   Gomory: 1
##   Cover: 1
##
## Explored 0 nodes (1582 simplex iterations) in 0.43 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 4.000000000000e+00, best bound 4.000000000000e+00, gap 0.0%
## Optimize a model with 1302 rows, 1275 columns and 5025 nonzeros
## Coefficient statistics:
##   Matrix range    [4e-02, 2e+03]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 8e+02]
## Found heuristic solution: objective 21
## Presolve time: 0.07s
## Presolved: 1302 rows, 1275 columns, 4983 nonzeros
## Variable types: 0 continuous, 1275 integer (1275 binary)
## Presolve removed 1 rows and 0 columns
## Presolved: 1301 rows, 1275 columns, 4980 nonzeros
```

```
##
## Presolve removed 1301 rows and 1275 columns
##
## Root relaxation: objective 3.289598e+00, 1099 iterations, 0.07 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0    3.28960    0  120   21.00000    3.28960  84.3%     -    0s
## H    0     0                       6.0000000    3.28960  45.2%     -    0s
## H    0     0                       4.0000000    3.28960  17.8%     -    0s
##
## Explored 0 nodes (1099 simplex iterations) in 0.18 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 4.000000000000e+00, best bound 4.000000000000e+00, gap 0.0%
```

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

```
# 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))
```

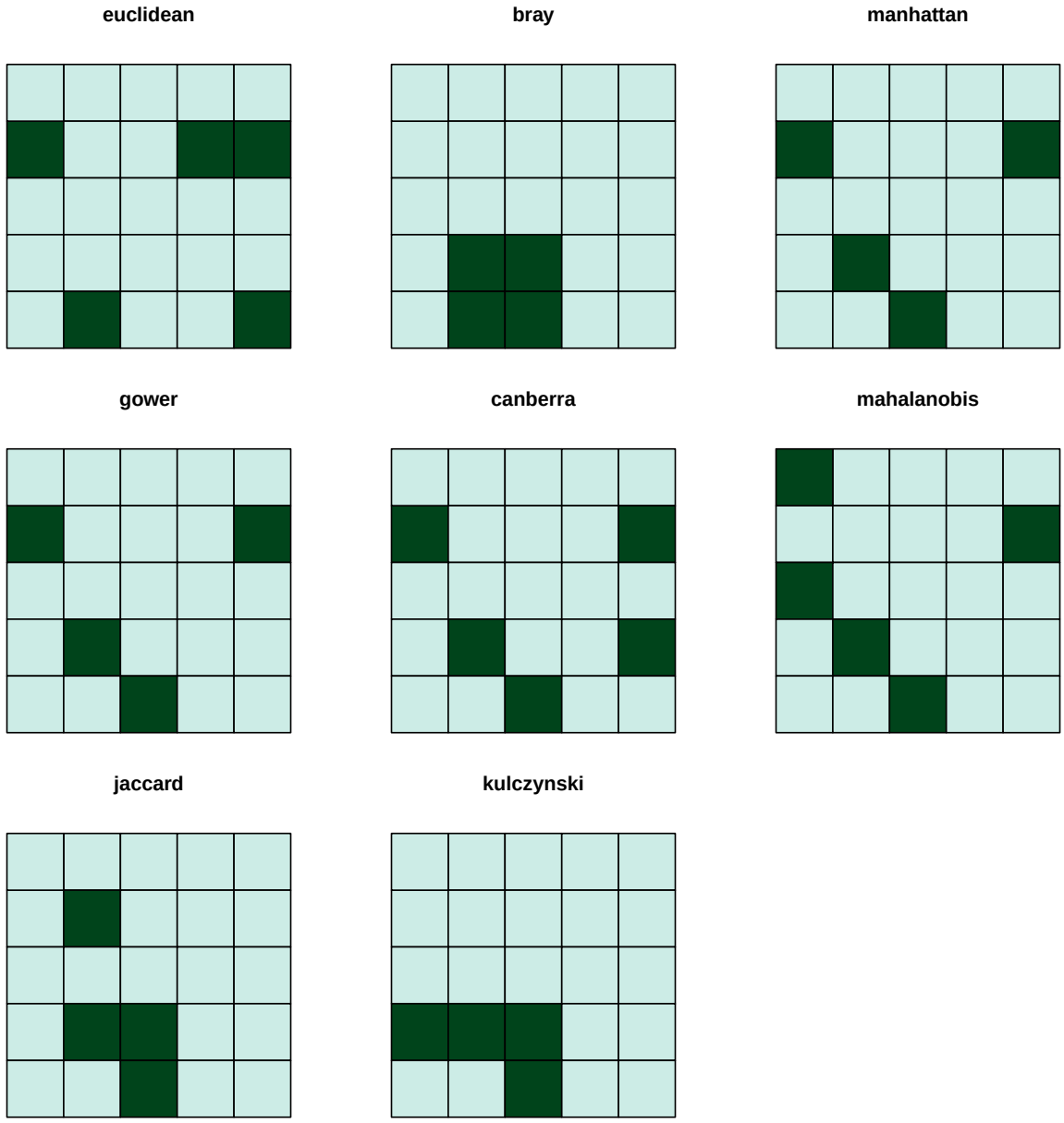


Figure 29: Prioritisations generated using different distance metrics. See Figure 3 for conventions.

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. Euclidean, Gower, and manhattan), whereas others select fewer planning units (eg. Bray-Curtis, Jaccard, and Kulczynski). Conservation planners should think carefully which distance metric is most apropriate for their attribute spaces.

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 kookuburra, 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 our purposes. 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 100km^2 grid, wherein the survey with the greatest number of repeat visits in each grid cell was chosen. To characterise the environmental variation at each survey location (site), climatic data (bio1, bio4, bio15, bio16, bio17) and classifications of the vegetation at the site were obtained. Occupancy-detection models (MacKenzie *et al.* 2002) were fit using Stan [Stan Development Team (2015); with parameters: adapt deta=0.9, maximum treedepth=20, chains=4 , warmup iterations=1000, total iterations=1500] using 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 using the best models identifed in each replicate, and then averaging them togeather. 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 acheived by masking out biogeographic regions where the species was not observed, and regions that did not have a neighbor where the species was observed. The maps were then resampled (10km^2 resolution) and cropped to the study area. The resulting maps are stored in the `cs_spp` object.

```
# load data
data(cs_spp)

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

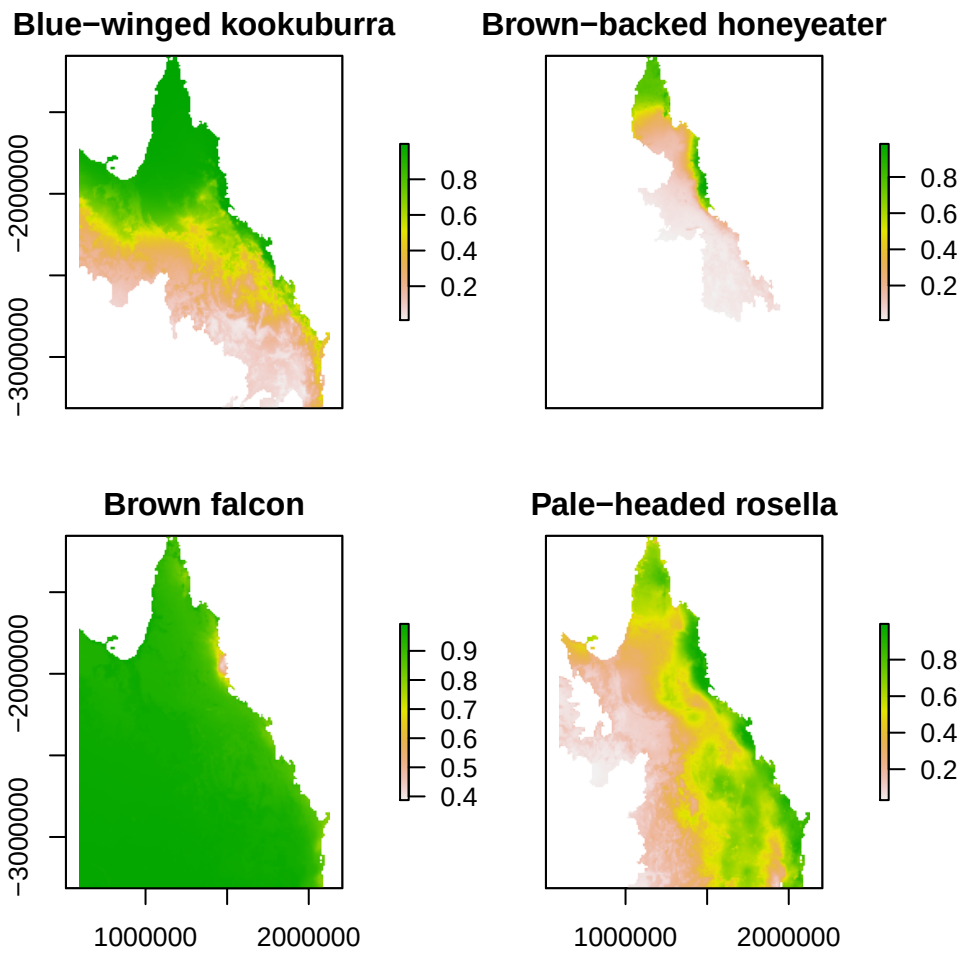


Figure 30: Distribution map for four Australian bird species. Pixels colors denote probability of occupancy.

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 colors 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'
)
```

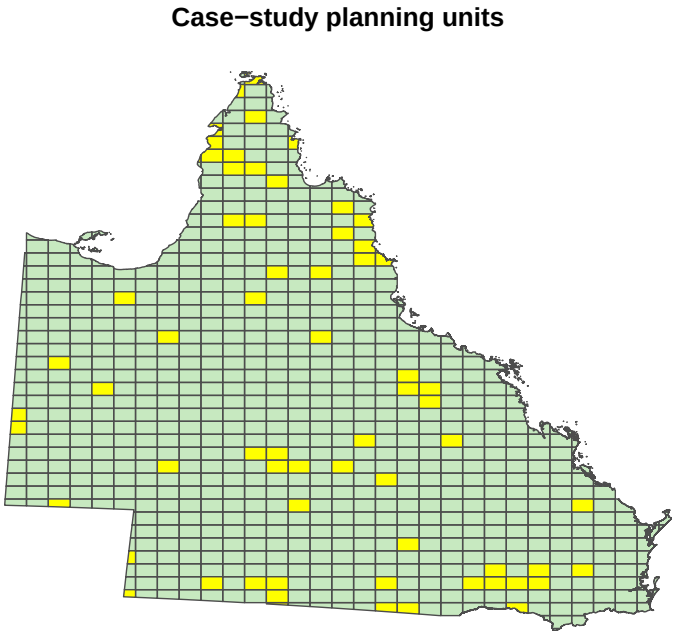


Figure 31: Planning units for the case-study examples. Yellow polygons represent planning units with more then 50% of their area already in existing reserves.

```
# 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 correspondance 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'))
```

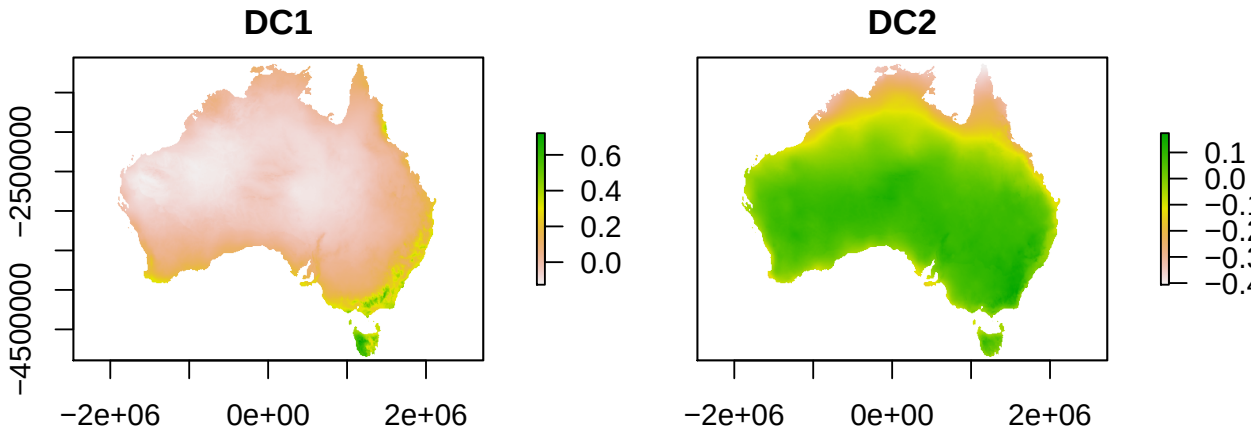


Figure 32: Broad-scale environmental variation across Australia. The variable DC1 describes the transition from wet and cool to dry and hot conditions. The variable DC2 describes the transition from wet and hot to dry and cool conditions.

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=0, 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
```

```r
# threshold probabilities to 0.5 for space calculations
cs_rs_amount <- prob.subset(cs_rs_amount, species=1:4, threshold=rep(0.5,4))
```

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

```
## Optimize a model with 4 rows, 762 columns and 1383 nonzeros
## Coefficient statistics:
##   Matrix range    [4e+02, 2e+04]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+05, 3e+06]
## Found heuristic solution: objective 161
## Presolve time: 0.01s
## Presolved: 4 rows, 762 columns, 1383 nonzeros
## Variable types: 0 continuous, 762 integer (762 binary)
## Presolved: 4 rows, 762 columns, 1383 nonzeros
##
##
## Root relaxation: objective 1.352823e+02, 726 iterations, 0.01 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0  135.28227    0    4  161.00000  135.28227  16.0%     -    0s
## H    0     0                     136.0000000  135.28227  0.53%     -    0s
##
## Explored 0 nodes (726 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%
```

```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         9836021          81320163          7726230
##   Connectivity_In_Fraction
## 1               0.09947189
```

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

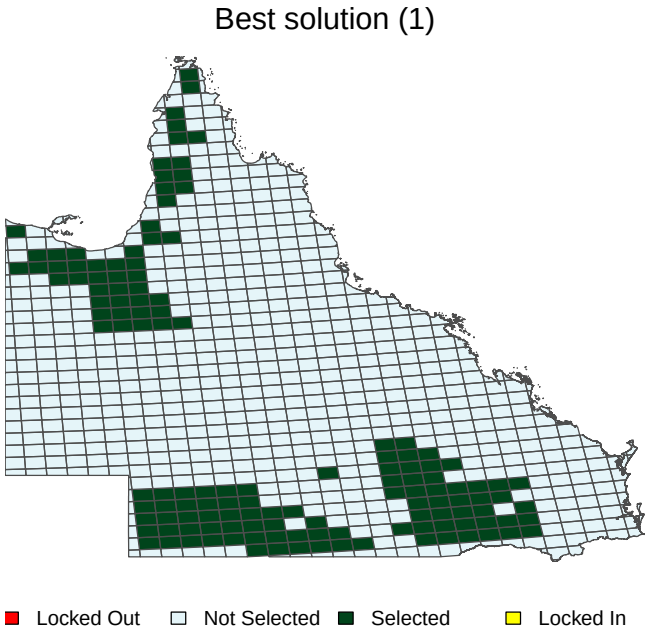


Figure 33: Multi-species prioritisation generated for four bird species using amount-based tagets (20%). See Figure 3 captions for conventions.

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 kookuburra')
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)
```

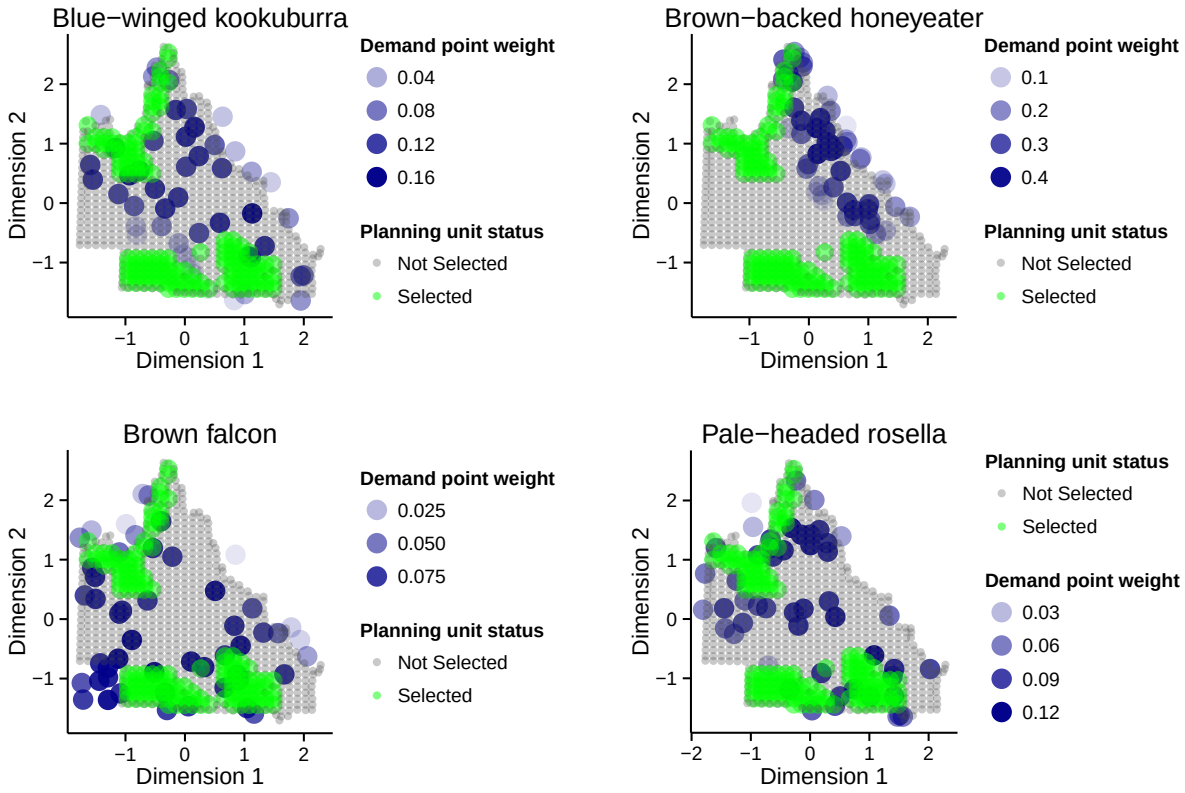


Figure 34: Distribution of amount-based prioritisation in the geographic attribute space. Points denote combinations of environmental conditions. Green and grey points represent planning unit selected for and not selected for prioritisation (respectively). Blue points denote demand points, and their size indicates their weighting.

```
# plot prioritisation in environmental attribute space
p1 <- space.plot(cs_rs_amount, 1, 1, main='Blue-winged kookuburra')
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)
```

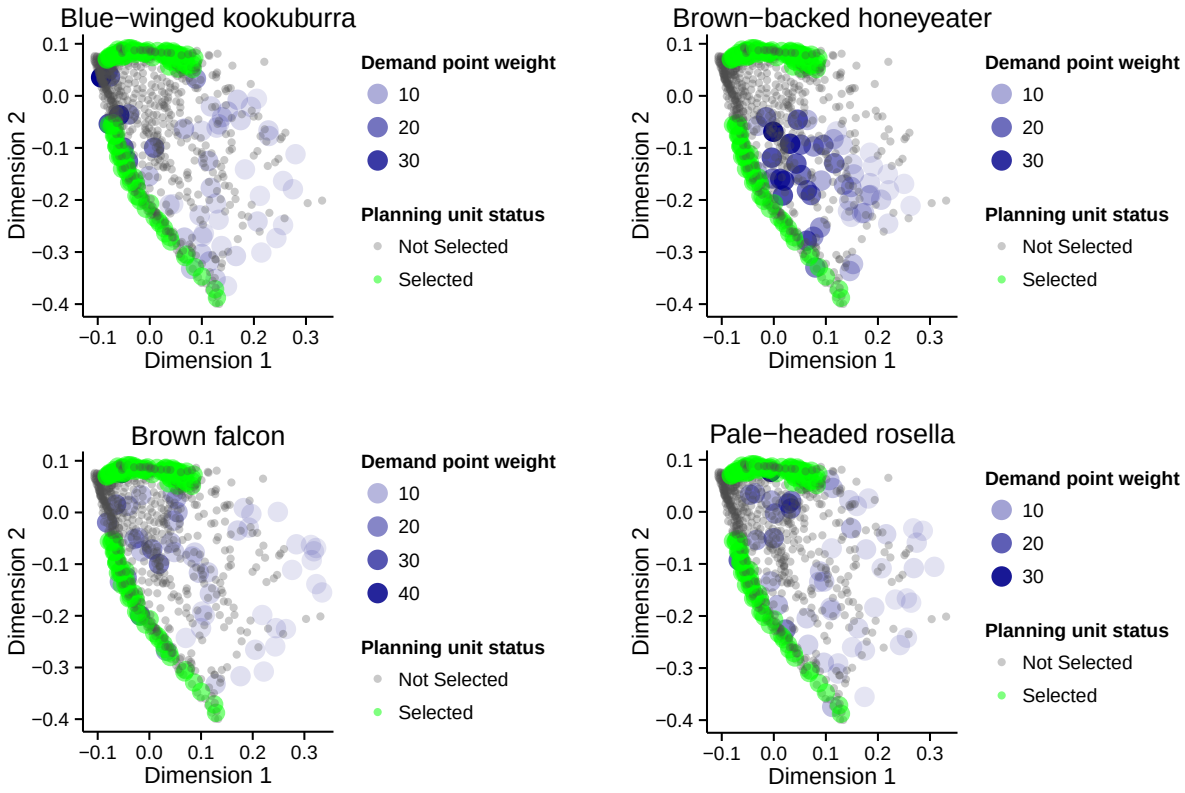


Figure 35: Distribution of amount-based prioritisation in the environmental attribute space. See Figure 34 caption for conventions.

Next, let's generate a prioritisation using amount- and space-based targets. This prioritisation will secure 95% of the species distribution in geographic and environmental space.

```
# make amount- and space-based prioritisation
cs_rs_space <- update(cs_rs_amount, space.target=0.85)
```

```
## Optimize a model with 138712 rows, 139062 columns and 554583 nonzeros
## Coefficient statistics:
##   Matrix range    [2e-05, 2e+05]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 3e+06]
## Presolve removed 0 rows and 336 columns (presolve time = 5s) ...
## Presolve removed 0 rows and 336 columns (presolve time = 10s) ...
## Presolve removed 337 rows and 338 columns (presolve time = 15s) ...
## Presolve removed 337 rows and 338 columns (presolve time = 20s) ...
## Presolve removed 337 rows and 338 columns (presolve time = 25s) ...
## Presolve removed 337 rows and 338 columns (presolve time = 31s) ...
## Presolve removed 337 rows and 338 columns (presolve time = 35s) ...
```

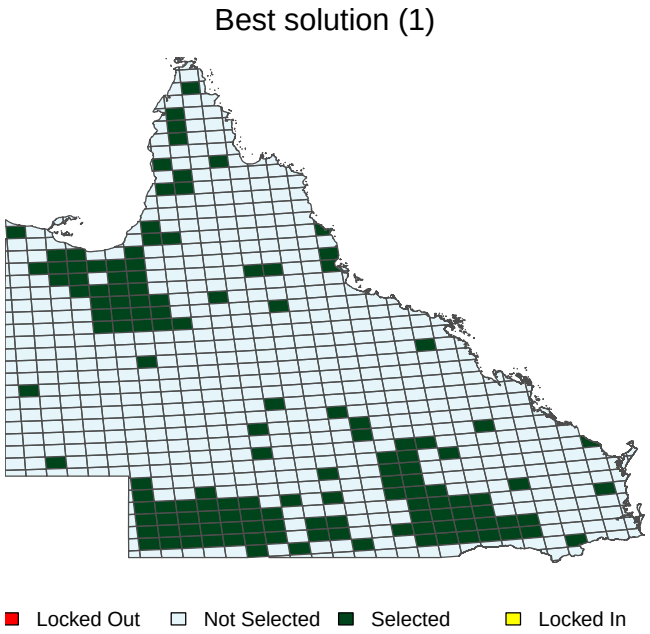
```
## Presolve removed 337 rows and 338 columns (presolve time = 40s) ...
## Presolve removed 337 rows and 338 columns (presolve time = 45s) ...
## Presolve removed 337 rows and 340 columns (presolve time = 50s) ...
## Presolve removed 337 rows and 340 columns (presolve time = 55s) ...
## Presolve removed 465 rows and 341 columns
## Presolve time: 56.35s
## Presolved: 138247 rows, 138721 columns, 679858 nonzeros
## Variable types: 0 continuous, 138721 integer (138721 binary)
## Presolve removed 1301 rows and 1 columns (presolve time = 5s) ...
## Presolve removed 1818 rows and 506 columns (presolve time = 10s) ...
## Presolve removed 1818 rows and 506 columns
## Presolved: 136429 rows, 138215 columns, 587498 nonzeros
##
## Presolve removed 128720 rows and 91427 columns
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    1.0000000e+00   1.441756e+01   2.832768e+07     70s
##      881    1.7075039e+02   0.000000e+00   3.376256e+03     70s
##    10032    1.3752945e+02   0.000000e+00   9.815941e+01     75s
##    12519    1.3739372e+02   0.000000e+00   0.000000e+00     77s
##    12519    1.3739372e+02   0.000000e+00   0.000000e+00     77s
##
## Root relaxation: objective 1.373937e+02, 12519 iterations, 18.56 seconds
## Total elapsed time = 81.61s
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0  137.39372    0  175          -  137.39372      -     -   82s
## H    0     0                     138.0000000  137.39372  0.44%     -   86s
##
## Explored 0 nodes (16128 simplex iterations) in 86.94 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 1.380000000000e+02, best bound 1.380000000000e+02, gap 0.0%
```

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

```
##   Run_Number Status Score Cost Planning_Units Connectivity_Total
## 1          1 MANUAL   138  138            138           98882414
##   Connectivity_In Connectivity_Edge Connectivity_Out
## 1         8281190          78249983         12351240
##   Connectivity_In_Fraction
```

```
## 1                0.08374786
```

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




```
# plot prioritisation in geographic attribute space
p1 <- space.plot(cs_rs_space, 1, 2, main='Blue-winged kookuburra')
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)
```

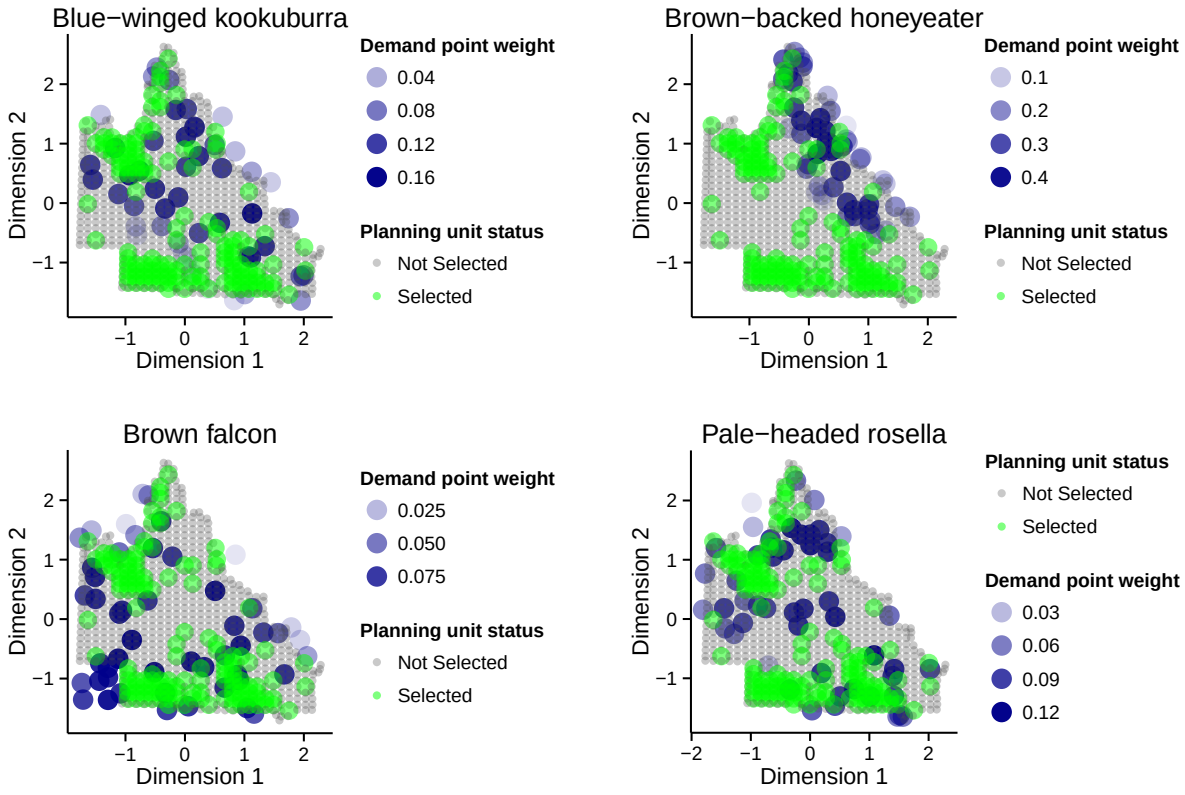


Figure 36: Distribution of the amount- and space-based prioritisation in the geographic attribute space. See Figure 34 caption for conventions.

```
# plot prioritisation in environmental attribute space
p1 <- space.plot(cs_rs_space, 1, 1, main='Blue-winged kookuburra')
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)
```

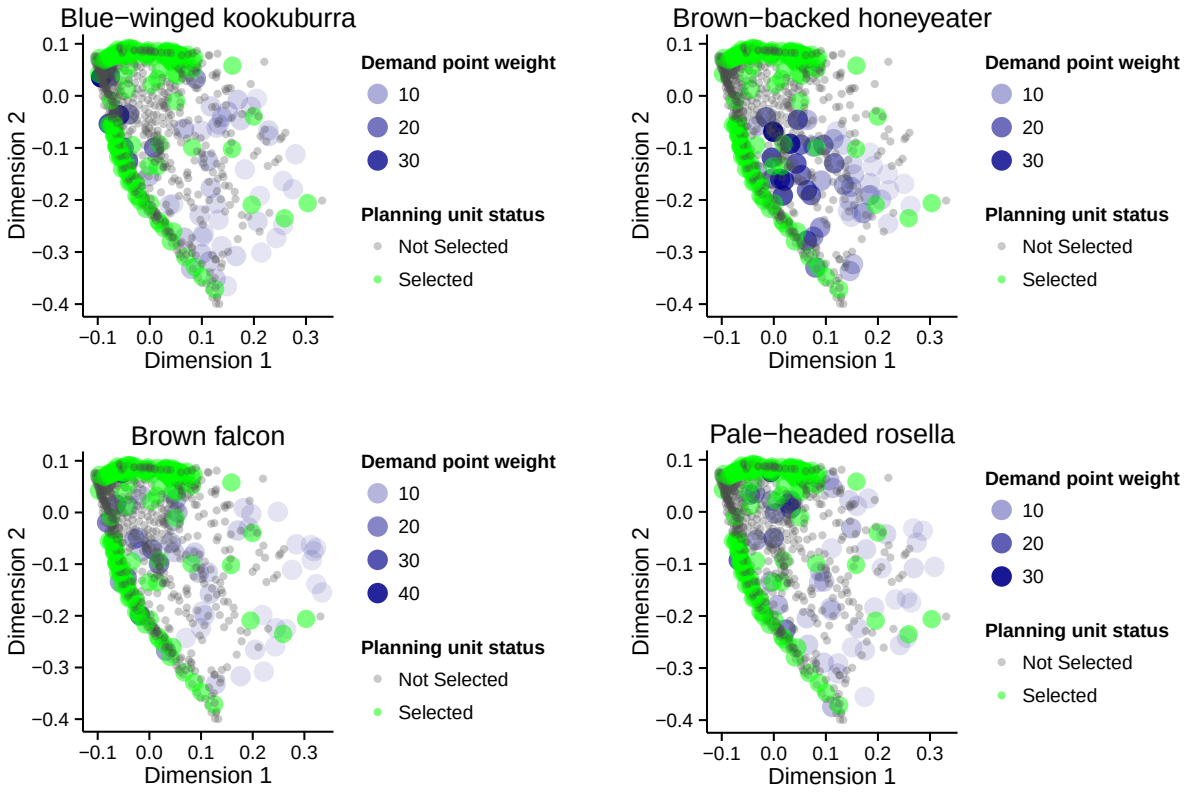


Figure 37: Distribution of the amount- and space-based prioritisation in the environmental attribute space. See Figure 34 caption for conventions.

Let's compare these prioritisations with Australia'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 Australia'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.

```
# load packages
library(dplyr)
library(ggplot2)

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

```r
# 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',
        'Australian 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')
```

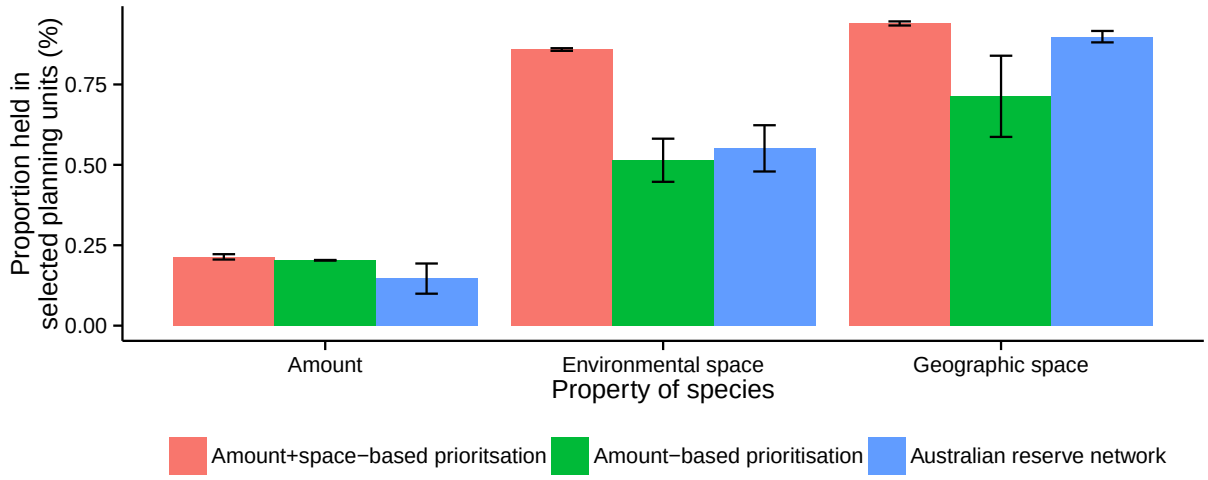


Figure 38: Prioritisations were generated using amount-based targets (20%), and with additional space-based targets (85%). These are compared to the Queensland reserve network. Data represent means and standard errors for the four species in each prioritisation.

These results suggest that prioritisations based just on amount-based targets can obtain a moderately representative sample of the species' geographic distribution and climatic niche. However, a much larger analysis is required to verify this.

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 138 targets. But what is the relationship between the space-targets and the number of planning units needed to satisfy the target?

Trade-off between representativeness and cost

To understand what the relationship between space-based targets and the size of a prioritisation, we will conduct another Pareto frontier analysis.

First, we will generate a suite of prioritisations using different space-based targets.

```
# set targets
space.targets <- round(seq(0.5,0.99,length.out=10),2)
```

```
# create empty list to store solutions
solutions <- list()

# generate solutions
for (i in seq_along(space.targets))
    solutions[[i]] <- update(cs_rs_space, space.target=rep(space.targets[i],4))
```

```
## Optimize a model with 138712 rows, 139062 columns and 554583 nonzeros
```

```
## Coefficient statistics:
##   Matrix range    [2e-05, 2e+05]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 3e+06]
## Found heuristic solution: objective 244
## Presolve removed 0 rows and 1 columns (presolve time = 5s) ...
## Presolve removed 0 rows and 1 columns (presolve time = 10s) ...
## Presolve removed 0 rows and 1 columns (presolve time = 15s) ...
## Presolve removed 0 rows and 1 columns (presolve time = 20s) ...
## Presolve removed 0 rows and 1 columns (presolve time = 25s) ...
## Presolve removed 0 rows and 1 columns (presolve time = 30s) ...
## Presolve removed 0 rows and 1 columns (presolve time = 35s) ...
## Presolve removed 1 rows and 1 columns
## Presolve time: 39.93s
## Presolved: 138711 rows, 139061 columns, 554191 nonzeros
## Variable types: 0 continuous, 139061 integer (139061 binary)
## Presolve removed 3 rows and 0 columns (presolve time = 5s) ...
## Presolve removed 3 rows and 0 columns (presolve time = 10s) ...
## Presolve removed 3 rows and 0 columns
## Presolved: 138708 rows, 139061 columns, 554182 nonzeros
##
## Presolve removed 138555 rows and 101162 columns
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    0.0000000e+00   1.776188e+00   1.238096e+07     51s
##     5701    1.3755832e+02   0.000000e+00   1.026125e+03     55s
##    13774    1.3570062e+02   0.000000e+00   5.232734e+01     60s
##    19570    1.3552109e+02   0.000000e+00   1.507814e+01     65s
##    25573    1.3545410e+02   0.000000e+00   2.040721e+01     70s
##    29209    1.3543053e+02   0.000000e+00   0.000000e+00     74s
##    29209    1.3543053e+02   0.000000e+00   0.000000e+00     74s
##
## Root relaxation: objective 1.354305e+02, 29209 iterations, 34.13 seconds
## Total elapsed time = 75.97s
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0  135.43053    0  153  244.00000  135.43053  44.5%     -   77s
## H    0     0                     136.0000000  135.43053  0.42%     -   78s
##
## Explored 0 nodes (34946 simplex iterations) in 78.58 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%
## Optimize a model with 138712 rows, 139062 columns and 554583 nonzeros
## Coefficient statistics:
##   Matrix range    [2e-05, 2e+05]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 3e+06]
## Found heuristic solution: objective 243
## Presolve removed 0 rows and 1 columns (presolve time = 5s) ...
## Presolve removed 0 rows and 1 columns (presolve time = 10s) ...
## Presolve removed 0 rows and 1 columns (presolve time = 15s) ...
## Presolve removed 0 rows and 1 columns (presolve time = 20s) ...
## Presolve removed 0 rows and 1 columns (presolve time = 25s) ...
## Presolve removed 0 rows and 1 columns (presolve time = 30s) ...
## Presolve removed 0 rows and 1 columns (presolve time = 35s) ...
## Presolve removed 1 rows and 1 columns
## Presolve time: 39.44s
## Presolved: 138711 rows, 139061 columns, 554203 nonzeros
## Variable types: 0 continuous, 139061 integer (139061 binary)
## Presolve removed 3 rows and 0 columns (presolve time = 5s) ...
## Presolve removed 3 rows and 0 columns (presolve time = 10s) ...
## Presolve removed 3 rows and 0 columns
## Presolved: 138708 rows, 139061 columns, 554194 nonzeros
##
## Presolve removed 138555 rows and 101162 columns
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    0.0000000e+00   2.475639e+00   1.238096e+07     50s
##       49    7.6200000e+02   0.000000e+00   7.620000e+02     50s
##     7735    1.3706481e+02   0.000000e+00   1.191943e+04     55s
##    14773    1.3581164e+02   0.000000e+00   7.118651e+01     60s
##    20776    1.3557144e+02   0.000000e+00   1.720960e+01     65s
##    25330    1.3550795e+02   0.000000e+00   8.198083e+00     70s
##    25947    1.3550463e+02   0.000000e+00   0.000000e+00     71s
##    25947    1.3550463e+02   0.000000e+00   0.000000e+00     71s
##
## Root relaxation: objective 1.355046e+02, 25947 iterations, 31.06 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0  135.50463    0  261  243.00000  135.50463  44.2%     -   74s
## H    0     0                     136.0000000  135.50463  0.36%     -   75s
##
## Explored 0 nodes (32576 simplex iterations) in 75.51 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%
## Optimize a model with 138712 rows, 139062 columns and 554583 nonzeros
## Coefficient statistics:
##   Matrix range    [2e-05, 2e+05]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 3e+06]
## Found heuristic solution: objective 241
## Presolve removed 0 rows and 2 columns (presolve time = 5s) ...
## Presolve removed 0 rows and 2 columns (presolve time = 10s) ...
## Presolve removed 0 rows and 2 columns (presolve time = 15s) ...
## Presolve removed 0 rows and 2 columns (presolve time = 20s) ...
## Presolve removed 0 rows and 2 columns (presolve time = 25s) ...
## Presolve removed 0 rows and 2 columns (presolve time = 30s) ...
## Presolve removed 0 rows and 2 columns (presolve time = 35s) ...
## Presolve removed 2 rows and 2 columns
## Presolve time: 39.34s
## Presolved: 138710 rows, 139060 columns, 554219 nonzeros
## Variable types: 0 continuous, 139060 integer (139060 binary)
## Presolve removed 3 rows and 0 columns (presolve time = 5s) ...
## Presolve removed 3 rows and 0 columns (presolve time = 10s) ...
## Presolve removed 3 rows and 0 columns
## Presolved: 138707 rows, 139060 columns, 554210 nonzeros
##
## Presolve removed 138554 rows and 101161 columns
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    0.0000000e+00   3.314981e+00   1.238096e+07     51s
##     7552    1.3728867e+02   0.000000e+00   5.266710e+02     55s
##    15211    1.3578520e+02   0.000000e+00   6.638495e+01     60s
##    21421    1.3563409e+02   0.000000e+00   1.542659e+01     65s
##    24655    1.3559843e+02   0.000000e+00   0.000000e+00     69s
##    24655    1.3559843e+02   0.000000e+00   0.000000e+00     69s
##
## Root relaxation: objective 1.355984e+02, 24655 iterations, 29.40 seconds
## Total elapsed time = 71.05s
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0  135.59843    0  277  241.00000  135.59843  43.7%     -   71s
## H    0     0                     136.0000000  135.59843  0.30%     -   72s
##
## Explored 0 nodes (30011 simplex iterations) in 72.21 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%
## Optimize a model with 138712 rows, 139062 columns and 554583 nonzeros
## Coefficient statistics:
##   Matrix range    [2e-05, 2e+05]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 3e+06]
## Found heuristic solution: objective 246
## Presolve removed 0 rows and 3 columns (presolve time = 5s) ...
## Presolve removed 0 rows and 3 columns (presolve time = 10s) ...
## Presolve removed 0 rows and 3 columns (presolve time = 15s) ...
## Presolve removed 0 rows and 3 columns (presolve time = 20s) ...
## Presolve removed 0 rows and 3 columns (presolve time = 25s) ...
## Presolve removed 0 rows and 3 columns (presolve time = 30s) ...
## Presolve removed 0 rows and 3 columns (presolve time = 35s) ...
## Presolve removed 3 rows and 3 columns
## Presolve time: 39.75s
## Presolved: 138709 rows, 139059 columns, 554259 nonzeros
## Variable types: 0 continuous, 139059 integer (139059 binary)
## Presolve removed 2 rows and 0 columns (presolve time = 5s) ...
## Presolve removed 2 rows and 0 columns (presolve time = 10s) ...
## Presolve removed 2 rows and 0 columns
## Presolved: 138707 rows, 139059 columns, 554253 nonzeros
##
## Presolve removed 138554 rows and 101160 columns
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    0.0000000e+00   4.014432e+00   1.238096e+07     51s
##     6083    1.3817395e+02   0.000000e+00   1.360006e+03     55s
##    14363    1.3581401e+02   0.000000e+00   7.105018e+01     60s
##    21401    1.3571600e+02   0.000000e+00   1.513089e+01     65s
##    24713    1.3569682e+02   0.000000e+00   5.539767e+00     70s
##    27346    1.3567660e+02   0.000000e+00   0.000000e+00     73s
##    27346    1.3567660e+02   0.000000e+00   0.000000e+00     73s
##
## Root relaxation: objective 1.356766e+02, 27346 iterations, 32.65 seconds
## Total elapsed time = 76.22s
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0  135.67660    0  273  246.00000  135.67660  44.8%     -   76s
## H    0     0                     136.0000000  135.67660  0.24%     -   77s
```

```
##
## Explored 0 nodes (34061 simplex iterations) in 77.57 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%
## Optimize a model with 138712 rows, 139062 columns and 554583 nonzeros
## Coefficient statistics:
##   Matrix range    [2e-05, 2e+05]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 3e+06]
## Presolve removed 0 rows and 4 columns (presolve time = 5s) ...
## Presolve removed 0 rows and 4 columns (presolve time = 10s) ...
## Presolve removed 4 rows and 4 columns (presolve time = 15s) ...
## Presolve removed 4 rows and 4 columns (presolve time = 20s) ...
## Presolve removed 4 rows and 4 columns (presolve time = 27s) ...
## Presolve removed 4 rows and 4 columns (presolve time = 30s) ...
## Presolve removed 4 rows and 4 columns (presolve time = 35s) ...
## Presolve removed 4 rows and 4 columns (presolve time = 40s) ...
## Presolve removed 4 rows and 4 columns (presolve time = 45s) ...
## Presolve removed 4 rows and 4 columns (presolve time = 50s) ...
## Presolve removed 4 rows and 4 columns
## Presolve time: 51.61s
## Presolved: 138708 rows, 139058 columns, 554762 nonzeros
## Variable types: 0 continuous, 139058 integer (139058 binary)
## Presolve removed 284 rows and 0 columns (presolve time = 5s) ...
## Presolve removed 285 rows and 0 columns (presolve time = 10s) ...
## Presolve removed 285 rows and 0 columns
## Presolved: 138423 rows, 139058 columns, 553908 nonzeros
##
## Presolve removed 138043 rows and 93667 columns
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    0.0000000e+00   6.582739e+00   2.032870e+07     64s
##     2372    1.4274301e+02   0.000000e+00   3.661417e+02     65s
##    11317    1.3677767e+02   0.000000e+00   5.576917e+03     70s
##    18527    1.3613225e+02   0.000000e+00   3.520753e+02     75s
##    24089    1.3600971e+02   0.000000e+00   2.207761e+01     80s
##    28827    1.3597563e+02   0.000000e+00   5.104039e+00     85s
##    33359    1.3596236e+02   0.000000e+00   1.382643e-02     90s
##    33452    1.3596211e+02   0.000000e+00   0.000000e+00     90s
##    33452    1.3596211e+02   0.000000e+00   0.000000e+00     91s
##
## Root relaxation: objective 1.359621e+02, 33452 iterations, 37.11 seconds
##
```

```
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0  135.96211    0  284          -  135.96211      -     -   94s
## H    0     0                     137.0000000  135.96211  0.76%     -   97s
##
## Explored 0 nodes (40673 simplex iterations) in 97.46 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%
## Optimize a model with 138712 rows, 139062 columns and 554583 nonzeros
## Coefficient statistics:
##   Matrix range    [2e-05, 2e+05]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 3e+06]
## Presolve removed 0 rows and 6 columns (presolve time = 5s) ...
## Presolve removed 0 rows and 6 columns (presolve time = 10s) ...
## Presolve removed 6 rows and 6 columns (presolve time = 15s) ...
## Presolve removed 6 rows and 6 columns (presolve time = 20s) ...
## Presolve removed 6 rows and 6 columns (presolve time = 27s) ...
## Presolve removed 6 rows and 6 columns (presolve time = 30s) ...
## Presolve removed 6 rows and 6 columns (presolve time = 35s) ...
## Presolve removed 6 rows and 6 columns (presolve time = 40s) ...
## Presolve removed 6 rows and 6 columns (presolve time = 45s) ...
## Presolve removed 6 rows and 6 columns (presolve time = 50s) ...
## Presolve removed 6 rows and 6 columns
## Presolve time: 51.25s
## Presolved: 138706 rows, 139056 columns, 559095 nonzeros
## Variable types: 0 continuous, 139056 integer (139056 binary)
## Presolve removed 291 rows and 0 columns (presolve time = 5s) ...
## Presolve removed 294 rows and 0 columns (presolve time = 10s) ...
## Presolve removed 294 rows and 0 columns
## Presolved: 138412 rows, 139056 columns, 557354 nonzeros
##
## Presolve removed 137344 rows and 93407 columns
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    0.0000000e+00   1.001136e+01   2.514007e+07     64s
##     2260    1.4578945e+02   0.000000e+00   6.256898e+02     65s
##     9699    1.3772141e+02   0.000000e+00   1.857923e+03     70s
##    14437    1.3672043e+02   0.000000e+00   3.882187e+02     75s
##    19793    1.3640832e+02   0.000000e+00   5.320490e+00     80s
##    20731    1.3639927e+02   0.000000e+00   0.000000e+00     81s
##    20731    1.3639927e+02   0.000000e+00   0.000000e+00     82s
```

```
##
## Root relaxation: objective 1.363993e+02, 20731 iterations, 28.17 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0  136.39927    0  251          -  136.39927      -     -   85s
## H    0     0                     137.0000000  136.39927  0.44%     -   88s
##
## Explored 0 nodes (25196 simplex iterations) in 88.85 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 1.370000000000e+02, best bound 1.370000000000e+02, gap 0.0%
## Optimize a model with 138712 rows, 139062 columns and 554583 nonzeros
## Coefficient statistics:
##   Matrix range    [2e-05, 2e+05]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 3e+06]
## Presolve removed 0 rows and 55 columns (presolve time = 5s) ...
## Presolve removed 0 rows and 55 columns (presolve time = 10s) ...
## Presolve removed 56 rows and 57 columns (presolve time = 15s) ...
## Presolve removed 56 rows and 57 columns (presolve time = 20s) ...
## Presolve removed 56 rows and 57 columns (presolve time = 27s) ...
## Presolve removed 56 rows and 57 columns (presolve time = 30s) ...
## Presolve removed 56 rows and 57 columns (presolve time = 35s) ...
## Presolve removed 56 rows and 57 columns (presolve time = 40s) ...
## Presolve removed 56 rows and 57 columns (presolve time = 45s) ...
## Presolve removed 56 rows and 57 columns (presolve time = 50s) ...
## Presolve removed 175 rows and 58 columns
## Presolve time: 52.09s
## Presolved: 138537 rows, 139004 columns, 779712 nonzeros
## Variable types: 0 continuous, 139004 integer (139004 binary)
## Presolve removed 1558 rows and 0 columns (presolve time = 5s) ...
## Presolve removed 2155 rows and 727 columns (presolve time = 10s) ...
## Presolve removed 2155 rows and 727 columns (presolve time = 15s) ...
## Presolve removed 2303 rows and 727 columns (presolve time = 20s) ...
## Presolve removed 2303 rows and 727 columns
## Presolved: 136234 rows, 138277 columns, 665588 nonzeros
##
## Presolve removed 127345 rows and 90986 columns (presolve time = 5s) ...
## Presolve removed 127345 rows and 90986 columns
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    1.0000000e+00   1.294547e+01   3.094533e+07     84s
```

```
##      174    7.6200000e+02   0.000000e+00   7.610000e+02     84s
##     2266    1.4570899e+02   0.000000e+00   1.674083e+03     85s
##    10429    1.3721757e+02   0.000000e+00   2.068107e+01     90s
##    13257    1.3718562e+02   0.000000e+00   0.000000e+00     92s
##    13257    1.3718562e+02   0.000000e+00   0.000000e+00     92s
##
## Root relaxation: objective 1.371856e+02, 13257 iterations, 38.09 seconds
## Total elapsed time = 97.04s
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0  137.18562    0  202          -  137.18562      -     -   97s
## H    0     0                     138.0000000  137.18562  0.59%     -  102s
##
## Explored 0 nodes (17262 simplex iterations) in 102.12 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 1.380000000000e+02, best bound 1.380000000000e+02, gap 0.0%
## Optimize a model with 138712 rows, 139062 columns and 554583 nonzeros
## Coefficient statistics:
##   Matrix range    [2e-05, 2e+05]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 3e+06]
## Presolve removed 0 rows and 1382 columns (presolve time = 5s) ...
## Presolve removed 0 rows and 1382 columns (presolve time = 10s) ...
## Presolve removed 1385 rows and 1388 columns (presolve time = 15s) ...
## Presolve removed 1385 rows and 1388 columns (presolve time = 20s) ...
## Presolve removed 1385 rows and 1388 columns (presolve time = 27s) ...
## Presolve removed 1385 rows and 1388 columns (presolve time = 30s) ...
## Presolve removed 1385 rows and 1388 columns (presolve time = 35s) ...
## Presolve removed 1385 rows and 1388 columns (presolve time = 40s) ...
## Presolve removed 1385 rows and 1389 columns (presolve time = 45s) ...
## Presolve removed 1385 rows and 1389 columns (presolve time = 50s) ...
## Presolve removed 1506 rows and 1392 columns
## Presolve time: 51.32s
## Presolved: 137206 rows, 137670 columns, 827467 nonzeros
## Variable types: 0 continuous, 137670 integer (137670 binary)
## Presolve removed 1590 rows and 0 columns (presolve time = 5s) ...
## Presolve removed 2322 rows and 727 columns (presolve time = 10s) ...
## Presolve removed 2322 rows and 727 columns
## Presolved: 134884 rows, 136943 columns, 547605 nonzeros
##
## Presolve removed 133056 rows and 91376 columns
##
## Root simplex log...
```

```
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    3.0000000e+00   1.375186e+01   3.106820e+07     69s
##     2874    1.4284029e+02   0.000000e+00   3.764104e+02     70s
##     9144    1.3778421e+02   0.000000e+00   0.000000e+00     75s
##     9144    1.3778421e+02   0.000000e+00   0.000000e+00     75s
##
## Root relaxation: objective 1.377842e+02, 9144 iterations, 21.15 seconds
## Total elapsed time = 77.38s
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0  137.78421    0  174          -  137.78421      -     -   80s
## H    0     0                     139.0000000  137.78421  0.87%     -   84s
##
## Explored 0 nodes (12550 simplex iterations) in 84.35 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 1.390000000000e+02, best bound 1.380000000000e+02, gap 0.7194%
## Optimize a model with 138712 rows, 139062 columns and 554583 nonzeros
## Coefficient statistics:
##   Matrix range    [2e-05, 2e+05]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [1e+00, 3e+06]
## Presolve removed 835 rows and 3017 columns (presolve time = 5s) ...
## Presolve removed 1131 rows and 3017 columns (presolve time = 10s) ...
## Presolve removed 1185 rows and 3017 columns (presolve time = 15s) ...
## Presolve removed 1204 rows and 3017 columns (presolve time = 20s) ...
## Presolve removed 1204 rows and 3017 columns (presolve time = 25s) ...
## Presolve removed 4221 rows and 3017 columns (presolve time = 30s) ...
## Presolve removed 4245 rows and 3029 columns (presolve time = 35s) ...
## Presolve removed 4252 rows and 3029 columns (presolve time = 40s) ...
## Presolve removed 4266 rows and 3029 columns (presolve time = 45s) ...
## Presolve removed 4279 rows and 3029 columns (presolve time = 50s) ...
## Presolve removed 4279 rows and 3029 columns (presolve time = 55s) ...
## Presolve removed 8645 rows and 108283 columns (presolve time = 60s) ...
## Presolve removed 108786 rows and 111984 columns (presolve time = 65s) ...
## Presolve removed 112303 rows and 111988 columns
## Presolve time: 69.15s
## Presolved: 26409 rows, 27074 columns, 110996 nonzeros
## Variable types: 0 continuous, 27074 integer (27074 binary)
## Presolve removed 287 rows and 7 columns
## Presolved: 26122 rows, 27067 columns, 109778 nonzeros
##
## Presolve removed 25192 rows and 16717 columns
```

```
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    7.0000000e+00   1.196139e+01   2.414752e+07     71s
##     9163    1.3934376e+02   0.000000e+00   0.000000e+00     73s
##     9163    1.3934376e+02   0.000000e+00   0.000000e+00     73s
##
## Root relaxation: objective 1.393438e+02, 9163 iterations, 2.03 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0  139.34376    0  184          -  139.34376      -     -   73s
## H    0     0                     140.0000000  139.34376  0.47%     -   73s
##
## Explored 0 nodes (9689 simplex iterations) in 73.61 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 1.400000000000e+02, best bound 1.400000000000e+02, gap 0.0%
## Optimize a model with 138712 rows, 139062 columns and 554583 nonzeros
## Coefficient statistics:
##   Matrix range    [2e-05, 2e+05]
##   Objective range [1e+00, 1e+00]
##   Bounds range    [1e+00, 1e+00]
##   RHS range       [4e-01, 3e+06]
## Presolve removed 2925 rows and 53107 columns (presolve time = 5s) ...
## Presolve removed 5422 rows and 53107 columns (presolve time = 10s) ...
## Presolve removed 7218 rows and 53107 columns (presolve time = 15s) ...
## Presolve removed 9568 rows and 53107 columns (presolve time = 20s) ...
## Presolve removed 11043 rows and 53148 columns (presolve time = 25s) ...
## Presolve removed 64231 rows and 53242 columns (presolve time = 30s) ...
## Presolve removed 64369 rows and 53242 columns (presolve time = 35s) ...
## Presolve removed 64461 rows and 53242 columns (presolve time = 40s) ...
## Presolve removed 64859 rows and 53242 columns (presolve time = 45s) ...
## Presolve removed 68581 rows and 53242 columns (presolve time = 50s) ...
## Presolve removed 79673 rows and 115047 columns (presolve time = 55s) ...
## Presolve removed 127207 rows and 115933 columns (presolve time = 60s) ...
## Presolve removed 127321 rows and 116825 columns (presolve time = 65s) ...
## Presolve removed 128293 rows and 118853 columns (presolve time = 70s) ...
## Presolve removed 128593 rows and 119808 columns (presolve time = 75s) ...
## Presolve removed 130237 rows and 123802 columns (presolve time = 80s) ...
## Presolve removed 130670 rows and 126169 columns (presolve time = 85s) ...
## Presolve removed 131016 rows and 127679 columns (presolve time = 90s) ...
## Presolve removed 131029 rows and 127692 columns (presolve time = 95s) ...
## Presolve removed 131029 rows and 127692 columns
## Presolve time: 96.77s
```

```
## Presolved: 7683 rows, 11370 columns, 63403 nonzeros
## Variable types: 0 continuous, 11370 integer (11370 binary)
## Presolve removed 156 rows and 5 columns
## Presolved: 7527 rows, 11365 columns, 62820 nonzeros
##
## Presolve removed 7117 rows and 9549 columns
##
## Root simplex log...
##
## Iteration    Objective       Primal Inf.    Dual Inf.      Time
##        0    2.5000000e+01   1.232060e+01   1.927857e+07     99s
##     3798    1.4651284e+02   0.000000e+00   0.000000e+00     99s
##     3798    1.4651284e+02   0.000000e+00   0.000000e+00     99s
##
## Root relaxation: objective 1.465128e+02, 3798 iterations, 0.52 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |     Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time
##
##      0     0  146.51284    0   72          -  146.51284      -     -   98s
## H    0     0                     148.0000000  146.51284  1.00%     -   99s
##
## Explored 0 nodes (3798 simplex iterations) in 99.14 seconds
## Thread count was 1 (of 2 available processors)
##
## Optimal solution found (tolerance 5.00e-02)
## Best objective 1.480000000000e+02, best bound 1.470000000000e+02, gap 0.6757%
```

Now, we will extract the proportion of geographic and envrionmental space held in each solution, as well as the number of planning units in the solution.

```
# load packages
library(plyr)
library(dplyr)

# make data.frame with results from each prioritisation
space_results <- ldply(solutions, .fun=function(x) {
    return(
        data.frame(
            n.planning.units=summary(x)$Planning_Units[1],
            space.target=space.target(x)[1],
            space.name=rep(c('Environmental space', 'Geographic space'),4),
            space.held=c(space.held(x))
        )
    )
}) %>% group_by(
    space.name,
    space.target
```

```
) %>% summarise(
    n.planning.units=first(n.planning.units),
    mean=mean(space.held),
    se=se(space.held)
)
```

Finally, let's visualise the Pareto frontiers.

```
# load package
library(ggplot2)

# plot Pareto frontiers
ggplot(aes(x=n.planning.units, y=mean, color=space.name),data=space_results) +
    geom_point() +
    geom_line() +
    geom_errorbar(aes(ymin=mean-se, ymax=mean+se), width=0.2) +
    theme_classic() +
    xlab('Cost of prioritisation (number of planning units in solution)') +
    ylab('Proportion of attribute space held (%)') +
    scale_color_manual(
        name='Attribute space',
        values=c('Environmental space'='#F8766D', 'Geographic space'='#00BFC4')
    )
```

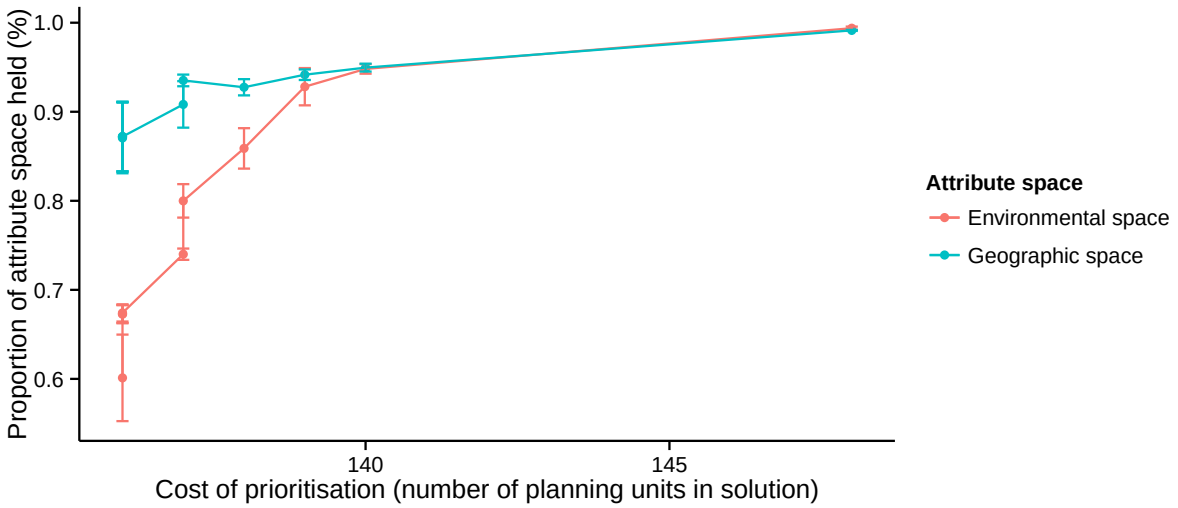


Figure 39: Data represent means and standard errors for the proportion of environmental and geographic space held in a prioritisation. Prioritsations were generated using BLM parameters: 0.50, 0.55, 0.61, 0.66, 0.72, 0.77, 0.83, 0.88, 0.94, and 0.99

Here we can see that additional planning units are need to maximise space representation. However, only 10 extra planning units are needed to have extremely good representation of environmental

and geographic space. It also appears that to fulfill environmental space-based targets, a large proportion of the geographic range of each species must also be captured. This result suggests that relying on geographic space-targets alone may fail to secure variation in environmental attribute spaces.

Implications and future directions

The `rapr` R package provides a unified approach to reserve selection. This R package decision maker with the tools to generate prioritisations that secure an adequate amount of a representative sample of biodiversity patterns and processes. Additionally, the decision maker can accomodate uncertainty in the distribution of features, and also ensure 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.

One of the key advantages of the `rapr` R package is that it is general enough that any any spatial variation could be considered an attribute space, regardless of whether this variation is intrinsic or extrinsic to the feature(s). As a consequence, in addition to biodiversity processes, this R package could be used to secure intra-specific (within feature) biodivery patterns. 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 (Ferrier et al. (2007)); gradients forests (GFs)], 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, this tools in this package can be misused, and could 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. The space-based targets are set as a proportion based on the level of representation if all the planning units are selected, and the worst prioritisation containing only one planning unit. As a consequence, if the decision maker uses an inapropriate set of spatial variables to construct an attribute space, or an inapropriate set of demand points, then the optimal solution based on this data will not be a cost-effective prioritisation. We therefore stress that decision makers 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 apropriate 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 random points within the polygon. A kernel is then fit to 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 biodiverity patterns and processes. Here, we developed the `rapr` R package to fill this void. By exploring the funciontality of this package using several simulated species, we found that including space-based targets can radically change a prioritisation for the simpliest 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

blank

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. (2015) Solving conservation planning problems with integer linear programming. *In prep.*

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.

Cowling, R. M., Pressey, R. L., Rouget, M., Lombard, A. T. (2003) A conservation plan for a global biodiversity hotspot - the cape floristic region, south africa. *Biological Conservation.* **112**, 191–216.

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.*

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.

Fernández, E., Landete, M. (2015) Fixed-charge facility location problems. In: *Location science* (eds G. Laporte, S. Nickel, & F. Saldanha da Gama) pp. 47–77 Springer International Publishing.

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.

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

Ponce-Reyes, R., Clegg, S. M., Carvalho, S. B., McDonald-Madden, E., Possingham, H. P. (2014) Geographical surrogates of genetic variation for selecting island populations for conservation. *Diversity & Distributions.* **20**, 640–651.

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.

Pressey, R. L., Watts, M. E., Barrett, T. W., Ridges, M. J. (2009) The C-Plan conservation planning system: origins, applications, and possible futures. In: *Spatial conservation prioritisation: Quantitative methods & computational tools* (eds A. Moilanen, K. A. Wilson, & H. Possingham) pp. 211–34 Oxford University Press, Oxford, UK.

Rouget, M., Cowling, R. M., Pressey, R. L., Richardson, D. M. (2003) Identifying spatial components of ecological and evolutionary processes for regional conservation planning in the cape floristic region, south africa. *Diversity & Distributions.* **9**, 191–210.

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.