Leipzig University
Faculty of Mathematics and Computer Science
Institute of Mathematics

Conservation Prioritization Problems and their Shadow Prices

A thesis presented for the degree of
Diplom Wirtschaftsmathematikerin

Leipzig, March 2015

Submitted by: Andrea Kaim

Supervisors: PD Dr. Anita Kripfganz (Leipzig University)

Prof. Dr. Hugh P. Possingham (The University of Queensland)

Abstract

Systematic conservation planning is an essential part of biodiversity preservation. In the context of conservation prioritization problems, the total cost of the entire reserve system is highly dependent on how big we set targets (e.g. 10% or 30%) for conservation features (e.g. species or habitats). Thus, it is of interest for conservation planners, how targets could be adjusted in a reasonable way in order to decrease total cost. The aim is to give a feature ranking based on their influence on the latter. Focusing on the minimum set coverage problem – an integer linear optimization problem (ILP) – this thesis presents a method to rank features according to their influence on total cost. Since the computation time is often too high to solve the ILP, its optimal solutions are approximated by the results of a linear optimization problem (LP). The shadow prices of the LP are used for the feature ranking which is compared to additional rankings. These are created by methods which used an ILP solver and the software Marxan which is based on a simulated annealing algorithm. The results showed that for the minimum set coverage problem shadow prices can be used to create an approximate feature ranking of impact on total cost. Furthermore many planning units selected for conservation by Marxan and the LP solver were the same. These results can be useful to improve Marxan. Additionally, the feature ranking provides a new supporting tool for decision makers in conservation planning.

Contents

1 Introduction

Our planet provides a diversity of habitats for millions of different species. Aside from their existence value, they provide very important ecosystem services such as food production, crop pollination and flood protection. Thus, it is important from an ecological, as well as an economic point of view, to conserve these species and areas. Cocks and Baird stated that "conservation reserves are areas of land intended to be retained in a relatively undisturbed state in the hope of ensuring the long-term survival of the biotic and abiotic entities they contain, e.g. species, communities, guilds, land units, land systems, geological formations." [8, p. 114]. This thesis is about the selection of conservation reserves for conserving biodiversity.

Generally there is a variety of sites which can be chosen as future protected areas but in most cases the selection is restricted by limited resources (e.g. a limited budget). Additionally, targets for the representation level of the considered features (e.g. 20% of the distribution of a species or habitat, or a minimum viable habitat area [11, 22]) must be met. Furthermore, the spatial relationships between sites and several other aspects should be taken into account.

The question of reserve design has been discussed by several scientists. Soulé and Simberloff, for example, emphasize the topic of population viability as an important criterion for designing reserve systems. They conclude that "nature reserves should be as large as possible, and there should be many of them. [...] For many species, it is likely that there must be vast areas, while for others, smaller sites may suffice as long as they are stringently protected and, in most instances, managed. If there is a target species, then the key criterion is habitat suitability." [23, p. 32]. In this context, Burgman et al. suggest a ranking of sites according to a habitat suitability index which could be used to calculate the total value of a habitat patch for species conservation [6]. The problem of the early work done in conservation planning is that it often lacks the consideration of social and economic constraints. However, later research realized this deficit and systematic approaches rose that suggested to build reserve networks "as efficiently as possible within a constrained area".[20, p. 291]

According to Cocks and Baird, the problem of selecting various sites in order to create a reserve system can be formulated as a mathematical programming problem. To this end, they provide a set of reserve guidelines such as: “ensure that only low cost sites are included in the suggested reserve system.” [8, p. 119]

There are many ways of formulating conservation prioritization problems depending on the decision makers’ objectives, restrictions and the considerations they would like to consider. A mathematical classification along with some examples will be provided in Chapter 2.

In this thesis, we will focus on the static versions of the so-called *minimum set coverage problem*, which is an integer linear programming problem (ILP), and its relaxation, the *land allocation problem*, a linear programming problem (LP). The definition of both optimization problems based on Moilanen et al. [18] will be given in Chapter 2. These problems represent the most basic approaches to formulating a conservation prioritization problem. Their objective is to minimize cost on the condition that the target for each feature is met. Given that targets influence the cost of the entire reserve system, it may be interesting for decision makers to know which feature requires what amount of the resources in order to adjust the targets in a reasonable way and thereby achieve lower total cost. Thus, the question emerges how we can create an appropriate feature ranking? Furthermore, we seek to understand, how targets can be changed (e.g. a slight increase in the minimum viable habitat area of a species) either without having any impact on total cost or by changing them drastically.

These are typical questions in the area of sensitivity analysis the answers to which require an optimal solution to the respective optimization problem to be found first. For this purpose, we will present different algorithms in Chapter 3: a branch-and-bound method for the resolution of relatively small ILPs and the dual bounded simplex algorithm for LPs. We will also introduce a simulated annealing algorithm, which is used by the conservation software Marxan and provides a heuristic approach for solving large ILPs.

Subsequently, methods for the sensitivity analysis of perturbations in the optimization problem’s right-hand side (RHS) will be given in Chapter 4. First, we cover linear optimization problems and introduce the so-called shadow prices, which under certain conditions can be used in order to describe changes in the objective function value due to perturbations in the RHS (i.e. the targets). For bigger perturbations, the use of the already calculated shadow prices might not be applicable anymore. In this case, linear parametric optimization techniques can be applied, which will be presented in section 4.1.2. We will also address the topic of degeneracy, which occurs when an optimization problem is overdetermined, i.e. there are more constraints than variables. An optimal degenerate solution implicates that instead of a unique shadow price, there might exist a whole range of shadow prices for a single optimal solution but only one of these can actually be used for our purposes. In general, the results of sensitivity analysis for LPs cannot be applied to ILPs. Therefore, the second part of the chapter will give an overview of approaches that can be used for integer linear optimization problems.

After we cover the theoretical background, we will develop a method that creates a feature ranking according to their influence on total cost of the reserve system (Chapter 5). We will then apply it to three real-world *minimum set coverage problems* and discuss the results in Chapter 6. Finally, the thesis will conclude with a brief outlook and suggestions for further research.

2 Conservation Prioritization Problems

In the 1970s, conservation biologists established the concept of refuge design in an attempt to explain species occurences in island-like habitats. The question emerged whether the creation of several small reserves or only a single large one would be the preferred way of preserving species richness. Until today, the concepts of conservation prioritization have been continuously developed in an attempt to describe the complexity of our world as realistically as possible. Formulating it in Hanski's words: "The goal is no longer a calculation that would just aim at selecting the next nature reserve to protect [...] The goal is [...] to apply statistical modelling techniques and numerical methods with the help of decision making theory to inform the rational allocation of resources that are available for conservation." [18, p. xvii]

In the introduction, we already used the terms 'feature', 'target' and 'cost', which are specified in the Marxan user's manual [11] as follows:

Features can be, for example, species, habitats or streams. A conservation prioritization problem can take a combination of different kind of features into account. Note that different features may have different units (e.g. hectares, number of occurrences, nests or length).

Targets define the amount, extent, number or degree of features that must be included in the solution. Each feature can have a different target, e.g. 20% of the overall representation of feature 1, 30% of feature 2. For conservation planners, it might be preferable to set high targets for rare and low targets for common conservation features. However, they must have the same units as given in the respective planning units.

Cost of a planning unit can be "any relative social, economic or ecological measure" [11, p. 46]. Monetary values, for example, can be obtained by using cost for purchasing land or "the opportunity cost of alternate land and sea uses that are incompatible with conservation" [11, p. 45 f.].

In the following, the mathematical classification of conservation prioritization problems is based on the corresponding chapter by Moilanen, Possingham and Polasky in [18]. According to the authors, conservation prioritization problems can be formulated as classical dynamic or static optimization problems consisting of one or multiple objective functions, control variables, constraints, state variables and state equations.

Symbol	Explanation
$i \in I = \{1, ..., n\}$	Index for sites
$j \in J = \{1, ..., m\}$	Index for features
$s \in \{0, 1\}$	Index for time
x_{is}	Dynamic indicator variable telling whether site i was chosen for evaluation at time s, $\mathbf{x}_s = (x_{1s}, ..., x_{ns})^T$
x_i	Static indicator variable telling whether site i was chosen for evaluation, $\mathbf{x} = (x_1, ..., x_n)^T$
a_{i0}	Action taken in site i at time 0, $\mathbf{a}_0 = (a_{10}, ..., a_{n0})^T$
a_i	Action taken in site i, $\mathbf{a} = (a_1, ..., a_n)^T$
c_i	Cost of site i, $\mathbf{c} = (c_1, ..., c_n)^T$
r_{ji}	Occurrence level of feature j in site i
$t_j \in \mathbb{R}_0^+$	Target representation level for feature j
B	Total conservation budget

Table 1: Notation.

They present three different approaches to the formulation of objective function and appropriate constraints: The first, the *minimum set coverage problem*, see (2.2) below, tries to achieve all conservation targets at minimum cost whereas the second, the *maximal coverage problem*, defined in (2.12), follows the objective to maximize the number of conservation targets having a restriction on the budget available. More generally, the objective could also be to maximize the conservation value that can be obtained with limited resources. This would be the third approach, called the *utility maximization problem*, introduced in (2.16).

In this thesis, we will discuss only static conservation prioritization problems. However, in order to guarantee a clear distinction between state and control variables, we will first introduce the *minimum set coverage problem* as a one-step dynamic problem and consequently, give then an alternative static definition, which is going to be the basis for the following chapters. A summary of the notation used in this chapter can be found in Table 1.

Definition 2.1: (*One-step dynamic minimum set coverage problem*)
Let $i \in I = \{1, \dots, n\}$ be the set of planning units and $\mathbf{x}_s$ the vector of indicator variables x_{is} stating whether site i has been selected for preservation

in time step $s \in \{0,1\}$ or not, i.e.

$$x_{is} = \begin{cases} 1, & \text{site } i \text{ selected} \\ 0, & \text{else} \end{cases} \quad \forall\, i, s.$$

Furthermore, let r_{ji} be the representation level of feature $j \in J = \{1, \ldots, m\}$ in planning unit i and let $t_j \in \mathbb{R}_0^+$ be the target for the respective feature. The vector $\mathbf{a}_0$ contains the actions a_{i0} taken in planning unit i at time step 0 and vector $\mathbf{c}$, the cost c_i of planning unit i. For a fixed inital vector $\mathbf{x}_0$, we define the *one-step dynamic minimum set coverage problem* as

$$\begin{aligned} \min_{\mathbf{a}_0, \mathbf{x}_1 \in \{0,1\}^n} \quad & \sum_{i \in I} c_i a_{i0} & \\ \text{s.t.} \quad & \sum_{i \in I} r_{ji} x_{i1} \geq t_j & \forall\, j \in J \\ & x_{i1} = 1 - (1 - x_{i0})(1 - a_{i0}) & \forall\, i \in I \\ & a_{i0}, x_{i1} \in \{0,1\} & \forall\, i \in I. \end{aligned} \tag{2.1}$$

Here, the objective is to minimize total cost of the reserve system provided that all targets are met. The x_{is} are the state variables describing the mathematical state of the reserve system, i.e. which sites are already conserved at time step 0 and which will be reserved at time step 1. The a_{i0} form the control variables. Those are the variables we can vary in order to find an optimal solution to the problem. In this case, an action is whether to select a planning unit for conservation or not, i.e.

$$a_{i0} = \begin{cases} 1, & \text{select site } i \\ 0, & \text{else} \end{cases} \quad \forall\, i. \tag{2.2}$$

The state equation

$$x_{i1} = 1 - (1 - x_{i0})(1 - a_{i0}) \;\forall\, i \in I$$

specifies the relation between state and control variables and the transition of the structure of the reserve system from time step 0 to 1. If site i has already been conserved or an according action has been taken in $s = 0$, x_{i1} obtains the value 1. Otherwise, it will is not considered as protected in $s = 1$, i.e. $x_{i1} = 0$.

It would also be possible to include the state variables of vector $\mathbf{x}_0$ in the minimization process in order to find an optimal starting vector. However, in the context of conservation science, it is more realistic to assume that the initial structure of the reserve system is given and that there is no opportunity to decide to protect a site in time step 0 or not.

We now define the static version of the *minimun set coverage problem.* Note that in this formulation, there is no clear distinction between control and state variables since both are conflated in the decision variable. On the one hand, the decision variable describes whether a site is part of the reserve system and on the other hand, it also states if a site is getting selected for conservation or not.

Definition 2.2: (*Static minimum set coverage problem*)
Let $\mathbf{x}$ be the vector of indicator variables x_i stating whether site i was selected for preservation or not. We then define the *static minimum set coverage problem* as

$$\begin{aligned} \min_{\mathbf{x}\in\{0,1\}^n} \quad & \sum_{i\in I} c_i x_i \\ \text{s.t.} \quad & \sum_{i\in I} r_{ji} x_i \geq t_j \quad \forall\, j \in J \\ & x_i \in \{0,1\} \qquad \forall\, i \in I. \end{aligned} \tag{2.3}$$

This is an integer optimization problem whose relaxation is called the *land allocation problem.* But before we introduce this problem, we will first examine the relationship of Definitions 2.1 and 2.2.

In Chapter 10.2 of [18], it is stated that the static problem can be thought of as a one-step dynamic problem. It appears that the two problems are not equivalent as shown in Example 2.1. However, if we assume that all sites are initially available for protection, we can prove that (2.1) and (2.2) lead to the same optimal solution.

Example 2.1: (*Non-equivalence of the dynamic and the static problem*)
Consider the *one-step dynamic minimum set coverage problem* (2.1) and let $\mathbf{x}_0 := (1, 0)$, i.e. the reserve system includes a site that is already protected at time step 0. For the dynamic problem, let r_{ji} and t_j be such that

$$\sum_{i\in I} r_{ji} x_{i1} \begin{cases} \geq t_j, \ \mathbf{x}_1 = \mathbb{1} \\ < t_j, \ \text{else} \end{cases} \quad \forall\, j \tag{2.4}$$

and for the static problem

$$\sum_{i\in I} r_{ji}x_i \begin{cases} \geq t_j, & \mathbf{x} = \mathbb{1} \\ < t_j, & \text{else} \end{cases} \quad \forall\, j. \tag{2.5}$$

Furthermore, we assume that costs are strictly positive, i.e. $\mathbf{c} > 0$.

Using the state equation of the dynamic problem, we get

$$x_{11} = 1 \text{ and } x_{21} = a_{20},$$

and entering these values in the objective function, we obtain

$$\begin{aligned} \min_{\mathbf{a}_0\in\{0,1\}^2} \sum_{i\in I} c_i a_{i0} &= \min_{\mathbf{a}_0\{0,1\}^2} (c_1 a_{10} + c_2 x_{21}) \\ &\overset{(2.4)}{=} \min_{\mathbf{a}_0\in\{0,1\}^2} (c_1 a_{10} + c_2) \\ &= c_2. \end{aligned} \tag{2.6}$$

The application of assumption (2.5) to the objective function of the static problem gives

$$\min_{\mathbf{x}\in\{0,1\}^2} \sum_{i\in I} c_i x_i = c_1 + c_2. \tag{2.7}$$

Comparing the results of (2.6) and (2.7), it is apparent that the static problem's objective function value is bigger than that of the dynamic problem. This is due to the fact that in the static case already protected sites are not considered as such. Therefore, if a site i is already reserved in time step 0 of the dynamic problem, i.e. $x_{i0} = 1$, and it is essential for the fulfilment of the constraints (meeting the targets), then it will be selected by the static problem as well. In this case, the protection of the site will cause cost in the present, whereas in the dynamic problem, these cost would have already been incurred in the past.

Corollary 2.1: The *one-step dynamic minimum set coverage problem* (2.1) can be formulated as a static problem by assuming that no planning units have already been selected for conservation, i.e. $\mathbf{x}_0 := 0$.

Proof: We first consider the dynamic problem. Using the state equation from (2.1), we get

$$\begin{aligned} x_{i1} &= 1 - (1 - x_{i0})(1 - a_{i0}) \\ &= 1 - (1 - a_{i0}) \\ &= a_{i0}. \end{aligned} \tag{2.8}$$

Now, we can reformulate the dynamic problem as

$$\begin{aligned} \min_{\mathbf{a}_0 \in \{0,1\}^n} \quad & \sum_{i \in I} c_i a_{i0} \\ \text{s.t.} \quad & \sum_{i \in I} r_{ji} a_{i0} \geq t_j \quad \forall\, j \in J \\ & a_{i0} \in \{0,1\} \qquad \forall\, i \in I. \end{aligned} \tag{2.9}$$

And with $x_i := a_{i0}$ for all $i \in I$, we obtain the static formulation as defined in Definition 2.2.

The other direction of proof starts with the static problem. Since we assume that $\mathbf{x}_0 := 0$, all sites are still available for conservation at time step 0. This means that the values of $\mathbf{x}$ depend on the actions taken in time step 0, $\mathbf{a}_0$, with a_{i0} as defined in (2.2). With the consideration of only one time step, both variables conflate. This means that

$$x_i = a_{i0}\ \forall i.$$

From (2.8), we can infer that

$$x_{i1} = x_i\ \forall i,$$

and by using this, the dynamic optimization problem has the form

$$\begin{aligned} \min_{\mathbf{x}_1 \in \{0,1\}} \quad & \sum_{i \in I} c_i x_{i1} \\ \text{s.t.} \quad & \sum_{i \in I} r_{ji} x_{i1} \geq t_j && \forall\, j \in J \\ & x_{i1} = 1 - (1 - x_{i0})(1 - a_{i0}) && \forall\, i \in I \\ & x_{i0} = 0, a_{i0}, x_{i1} \in \{0,1\} && \forall\, i \in I, \end{aligned} \tag{2.10}$$

which proves the assertion. $\square$

If we relax the *static minimum set coverage problem* by allowing its variable x_i to take continuous values, we obtain the *land allocation problem*:

Definition 2.3: (*Land allocation problem*)
The land allocation problem is a continuous optimization problem and can be written as

$$\begin{aligned} \min_{\mathbf{x}\in[0,1]^n} \quad & \sum_{i\in I} c_i x_i \\ \text{s.t.} \quad & \sum_{i\in I} r_{ji} x_i \geq t_j \quad \forall\, j \in J \\ & 0 \leq x_i \leq 1 \qquad \forall\, i \in I. \end{aligned} \tag{2.11}$$

Since $x_i \in [0,1]$ for all i, it is possible to select only fractions of sites. Imagine the case when a site consists of more than one habitat, and species living in this site are not homogeneously distributed, but prefer one habitat over another. A solution to the optimization problem might be to protect only a fraction of this site. However, in doing so, we would not be sure whether the fraction of the site that we protect actually contains the species or habitat we are interested in conserving. Therefore, it may be possible that some features would actually not be protected even though they would have been mathematically. Thus, the *minimum set coverage problem* may be a more realistic approach for real-world problems, but usually it is also more difficult to solve since it is an integer problem. Note that in the case species and habitats being homogeneously distributed through each site, the land allocation problem is valid for meeting targets not only theoretically, but also in practice.

As mentioned previously, another approach is to maximize the number of conservation targets subject to the condition that total cost should not exceed a certain budget. The *maximal coverage problem* is such a problem type and is defined below.

Definition 2.4: (*Maximal coverage problem*)
Let

$$y_j(\mathbf{x}) := \sum_{i\in I} r_{ji} x_i \;\; \forall\, j \in J$$

be the amount of protected individuals of feature j. Given a conservation budget B (e.g. money, trained personnel or time), the maximal coverage problem can be formulated as

$$\begin{aligned}
\max_{\mathbf{x}\in[0,1]^n} \quad & \sum_{j\in J}\theta(y_j(\mathbf{x}) - t_j) \\
\text{s.t.} \quad & \sum_{i\in I} c_i x_i \le B \qquad\qquad (2.12)\\
& x_i \in [0,1] \qquad \forall\, i \in I,
\end{aligned}$$

with

$$\theta(y_j(\mathbf{x}) - t_j) = \begin{cases} 1, \; y_j(\mathbf{x}) \ge t_j \\ 0, \; y_j(\mathbf{x}) < t_j \end{cases} \quad \forall\, j \in J.$$

The shifted unit step function θ takes the value 1 if the target of feature j is met and 0 otherwise. Solving this type of problem can be done by reformulating it into a mixed integer programming problem using an auxiliary binary variable, which will be shown in Corollary 2.2. Another way would be approximating the step function with a smooth function, even though this approach would lead to an approximate instead of a direct solution.

Corollary 2.2: The *maximal coverage problem* can be formulated as a mixed integer problem of the form

$$\begin{aligned}
\max_{\mathbf{z}\in\{0,1\}^n, \mathbf{x}\in[0,1]^n} \quad & \sum_{j\in J} z_j && && (2.13\text{a})\\
\text{s.t.} \quad & \sum_{i\in I} r_{ji} x_i \ge t_j z_j && \forall\, j \in J && (2.13\text{b})\\
& \sum_{i\in I} r_{ji} x_i \le t_j + M z_j && \forall\, j \in J && (2.13\text{c})\\
& \sum_{i\in I} c_i x_i \le B && && \\
& x_i \in [0,1], z_j \in \{0,1\} && \forall\, i \in I, j \in J &&
\end{aligned}$$

with z_j being an auxiliary binary variable and $M \gg 0$.

Proof: Consider the *maximal coverage problem* as it has been defined in (2.12). In order to obtain the objective function (2.13a), we define

$$z_j := \theta(y_j(\mathbf{x}) - t_j) = \begin{cases} 1, \ y_j(\mathbf{x}) \geq t_j \\ 0, \ y_j(\mathbf{x}) < t_j \end{cases} \quad \forall\, j \in J. \tag{2.14}$$

We can also add constraints (2.13b) and (2.13c) because if $z_j = 1$ for a $j \in J$, then

$$\sum_{i \in I} r_{ji} x_i \geq t_j \Leftrightarrow \sum_{i \in I} r_{ji} x_i \geq t_j z_j \quad \forall j$$

and

$$\sum_{i \in I} r_{ji} x_i \leq t_j + M z_j$$

would be redundant for all j with M big enough. And if $z_j = 0$ for a $j \in J$ then (2.13b) would be redundant since $r_{ji}, x_i \geq 0$ and constraint (2.13c) would be met, given that per definition of z_j, it holds that

$$\sum_{i \in I} r_{ji} x_i < t_j \ \forall j.$$

Thus, we obtain the problem formulation as given in (2.13).

Now consider the mixed integer problem formulation (2.13). Using the same reasoning as above, we can conclude that for

$$\sum_{i \in I} r_{ji} x_i > t_j$$

z_j must be 1 and for

$$\sum_{i \in I} r_{ji} x_i < t_j$$

it must take the value 0. In case

$$\sum_{i \in I} r_{ji} x_i = t_j$$

we have to distinguish the two cases in which t_j is equal to or greater than 0. It is obvious that if $t_j > 0$, then $z_j > 0$. But for $t_j = 0$, both $z = 0$ and $z = 1$ would fulfil the inequality if it were not for the objective function which always prefers $z = 1$ over $z = 0$. Therefore, we can conclude that for the currently discussed case, z must be equal to 1.

Defining z_j as in (2.14), we can eliminate the corresponding constraints (2.13a) and (2.13c) from the optimization problem and obtain the same problem formulation as in (2.12). $\square$

In the one-step dynamic formulation of the *minimum set coverage problem*, a_{is} has been the action taken in planning unit i at time step $s \in \{0, 1\}$. Accordingly, a_i is the respective variable of the static problem which we have addressed by the variable name x_i in (2.2) - (2.12). So far, we have considered only one action that can be taken in a planning unit (e.g. do we select the planning unit or not?) which is common for simple problem formulations. In spatial conservation planning, actions are usually purchasing land but they could also include actions such control of invasive species or fire management. In fact, a_i does not necessarily have to be binary or restricted to the value of 1 as in the case of x_i. It may also take continuous or discrete values. For example, if the planning units are allocated by different types of land use, the control variable can look like this:

$$a_i \in \{1 = \text{agriculture}, 2 = \text{forestry}, 3 = \text{developed}, 4 = \text{natural}\}\ \forall i. \quad (2.15)$$

This aspect is considered, for example, in the conservation planning software Marxan with Zones [29, 21]. Having the opportunity of choosing among several actions can make the optimization problem more complex since actions taken in one site can also have an influence on another planning unit. However, in scenario evaluation, one has the choice amongst different policies, each consisting of several actions, which leads to fewer complexities.

Additionally, cost c_i and the occurrence levels of the features r_{ji} can also be defined as functions. They can be depend on the actions taken in one site a_i, e.g. $r_{ji}(a_i)$, or even on the actions taken in the entire landscape $\mathbf{a}$, for instance $r_{ji}(\mathbf{a})$. Environmental conditions or the occurrence levels of other features in neighbouring sites might also influence the representation level of a feature in a certain planning unit. Therefore, it may also be useful to consider connectivity-related factors. And since in practice, it is often difficult to obtain absolute values for r_{ji}, statistical estimates are used frequently. The problem's structure would become even more complex if we did not assume a static, but a dynamic model with time-varying occurrence levels.

An example of a multiple-action conservation prioritization problem using the *utility maximization* approach mentioned before is given in Example 2.2. Note that in this example, we are describing a static optimization problem with a_i being the decision variables for all i. The $\bar{x}_i$ only represent the initial state of the reserve system. This means that $\bar{\mathbf{x}}$ is given and the optimization is going to be over the actions $\mathbf{a}$ only.

Example 2.2: (*Utility maximization problem*)
Let $\bar{\mathbf{x}} = (\bar{x}_1, ..., \bar{x}_n)$ be the structure of the reserve network with $\bar{x}_i \in \{0, 1\}$ for all i and $\mathbf{a} = (a_1, ..., a_n)$ is the vector of actions taken through the landscape, with a_i taking values in the set of possible actions $\{1, 2, 3, 4\}$. Let the level of species representation be a function $r_{ji}(\mathbf{a})$ which means that the occurrence level of a feature in a site is dependent on the actions taken in the entire landscape. Analogously let the cost $c_i(a_i)$ of site i be a function of the action taken in the respective site. Furthermore, let f_j be a benefit function transforming the representation of feature j to conservation value. Then, the following problem is called a *multiple-action utility maximization problem*:

$$\begin{aligned} \max_{\mathbf{a}\in\{1,\dots,4\}^n} \quad & \sum_{j\in J} f_j \left(\sum_{i\in I} \bar{x}_i r_{ji}(\mathbf{a}) \right) \\ \text{s.t.} \quad & \sum_{i\in I} c_i(a_i) \le B \\ & a_i \in \{1,2,3,4\} \qquad \forall\, i \in I. \end{aligned} \tag{2.16}$$

Here, $\bar{\mathbf{x}}$ is not a variable, but has fixed values and the decision variable is $\mathbf{a}$. As in (2.12), there is a budget constraint. However, in this problem formulation, the cost are not influenced by the structure of the reserve system itself, but by the actions taken in the respective sites. In addition to a budget constraint, there are also other possible constraints, such as political or social ones that can have an influence on the feasibility of certain actions.

Depending on its formulation, a conservation prioritization problem may turn out to be non-linear. For simplicity, we will focus in on the *minimum set coverage problem* (2.2) and its relaxation (2.11) only, assuming that all variables and constraints are linear and therefore, defining the *minimum set coverage problem* as a 0-1 multidimensional knapsack problem. Now we can consider it as an integer linear optimization problem (ILP) and the *land allocation problem* as a linear optimization problem (LP), respectively.

In order to illustrate the optimization problems and the techniques presented in this thesis, we introduce a fictitious data set in the example below.

Example 2.3: (*Illustrative data set*)
Table 2 shows the represenation levels of two species in three sites and their targets. Furthermore, costs for each planning unit are given in the last column.

Site	**Occurence level species 1**	**Occurence level species 2**	**Cost in $**
1	3	6	400
2	0	4	300
3	4	3	600
Target	3	7	-

Table 2: Illustrative data set.

Using the information given above, we obtain the *minimum set coverage* problem

$$\begin{aligned} \min_{\mathbf{x}\in\{0,1\}^3} \quad & 400x_1 + 300x_2 + 600x_3 \\ \text{s.t.} \quad & 3x_1 + 4x_3 \geq 3 \\ & 6x_1 + 4x_2 + 3x_3 \geq 7 \\ & x_i \in \{0,1\} \ \forall\, i \in I. \end{aligned} \tag{2.17}$$

It can be seen that the optimal solution of this problem is selecting sites 1 and 2 for conservation, i.e. $\mathbf{x}^*_{ILP} = (1,1,0)^T$, protecting 3 individuals of species 1 and 10 individuals of species 2. Thereby, we even exceed the target for species 2 obtaining minimum cost at \$700.

Figure 1 shows a three-dimensional plot of this optimization problem. The light blue surface illustrates the first, and the dark blue the second constraint. Thus, the polyhedron, which is being generated by these constraints and the variable boundaries, forms the set of feasible solutions. The red surface shows the objective function at a total cost level of \$700. One can imagine that during the optimization process, the plane moves from the upper part of the polyhedron to the lower one, since we are solving a minimization problem. Therefore, the optimal feasible solution lies at the lowest vertex of the considered polyhedron, right at the intersection of the red plane with the lowest feasible point. In our case, this happens at point $P_1 = \mathbf{x}^*_{ILP}$.

As it can be seen in Figure 1, $P_2 = (1, 0.25, 0)$ is an even lower vertex than P_1. But since $x_2 = 0.25$ is not an integer solution, it is only feasible for the relaxed version of the ILP, i.e. the *land allocation problem*. Here we allow x_i to take any value in the interval $[0,1]$ for all i. More precisely, this means that in this case, it is indeed allowed to protect only a quarter of site 2. Hence, $\mathbf{x}^*_{LP} = P_2$ is the optimal solution of the LP with total cost of \$475.

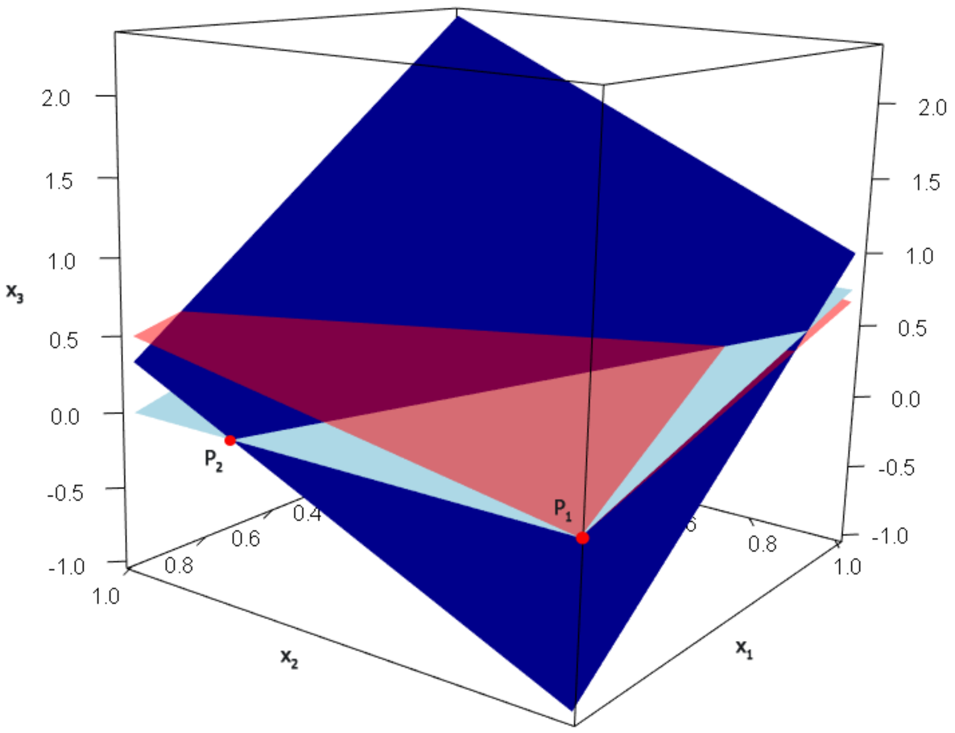


Figure 1: Three-dimensional plot of the illustrative data set with P_1 being the optimal integer solution and P_2 the optimal solution of the relaxed problem.

In the example above, we solved the optimization problem graphically. For real-world conservation prioritization problems, we have to come back to algorithms, three of which are presented here and are applicable to the *minimum set coverage problem* and the *land allocation problem* respectively.

3 Solving Conservation Prioritization Problems

Depending on their mathematical classifications, there exist many different approaches to solving optimization problems. Exact resolution methods such as branch and bound or the simplex algorithm find optimal solutions, but can be very time-consuming when it comes to high-dimensional problems with many constraints [9]. In contrast to this, heuristic optimization techniques do not guarantee to find optimal, but good feasible solutions in reasonable computation times [18].

The *minimum set coverage problem* from Chapter 2 is considered to be a so-called NP-Complete optimization problem. In particular, "optimal solution times for problems of this category rise faster than linearly with the size of the problem, where the problem size is the number of planning units and number of conservation features" [18]. Practically, it is not often possible to solve this problem type with an exact method and heuristics such as the simulated annealing algorithm must be applied.

The *land allocation problem*, however, can be solved quickly by exact approaches, for example with the simplex algorithm.

In this chapter we are going to present two exact resolution methods: first giving a brief introduction to the branch-and-bound approach, and then focusing more on the details of the dual bounded simplex algorithm. Furthermore, we will explain simulated annealing and here in particular, the underlying algorithm of the software Marxan which is widely used for solving conservation prioritization problems [18].

3.1 Branch & Bound

Branch and bound methods can be used to find integer solutions of a given ILP such as *minimum set coverage problems* with a small underlying data set. Without the integer constraints, optimal solutions can be found at the vertices of a feasible set, whereas in integer programming, this is not always the case. Optimal solutions can be far away from any vertex and may even lie amidst the search space and not at its border.

The idea of branch and bound is to divide an ILP into sub-problems and use the lower bounds of their objective function values to reduce the size of the search space. There are different approaches for the branching and bounding of an optimization problem. In this section, we will present Dakin's method for minimization problems based on the description in Domschke and Drexl 2005 [9], modified for the problem type considered here.

Branching divides the current initial problem P_0 into k sub-problems $P_1, ..., P_k$ such that

$$X(P_0) = \bigcup_{i=1}^{k} X(P_i) \quad \text{and} \quad X(P_i) \cap X(P_j) = \emptyset \; \forall i \neq j,$$

with $X(P_i)$ being the set of feasible solutions of problem P_i.

Thereby, we create a search tree whose individual nodes must be checked for optimal solution candidates. In order to restrain the branching process, the algorithm provides rules for the exclusion of sub-problems from the search list (bounding).

Therefore, we set the initial global upper bound of the objective function value to $\overline{F} := +\infty$ or find a better value with the aid of a heuristic approach. In order to evaluate if a sub-problem P_i must be branched again, we must find its local lower bound $\underline{F}_i$ by solving the relaxed problem $P_i^{'}$ and comparing it to the incumbent $\overline{F}$. The relaxation can be solved by the simplex algorithm, for instance.

If one of the following cases applies to the optimal objective function value of $P_i^{'}$, the sub-problem will be removed from the search list:

(i) ($\underline{F}_i \geq \overline{F}$): Since $X(P_i) \subseteq X(P_i^{'}) \;\; \forall i$, the optimal solution of the sub-problem cannot be better than the current best solution.

(ii) ($\underline{F}_i < \overline{F}$ and the optimal solution of $P_i^{'}$ is feasible for P_i): A new global best solution has been found. Save the solution and set $\overline{F} := \underline{F}_i$.

(iii) ($X(P_i^{'}) = \emptyset$): $X(P_i) = \emptyset$.

In all other cases, the problem must be branched and the sub-problems added to the search list. One possibility for branching gives the following theorem.

Theorem 3.1: (*Branching*, [9])
Let $\mathbf{x}_i^*$ be an optimal solution of $P_i^{'}$. Let x_j^* be a non-integer component of $\mathbf{x}_i^*$. Then, the sub-problems of P_i can be formed by

$$P_{i_1} := \{\mathbf{x}_i \in X(P_i) \mid x_j \leq \lfloor x_j^* \rfloor\} \text{ and } P_{i_2} := \{\mathbf{x}_i \in X(P_i) \mid x_j \geq \lceil x_j^* \rceil\}.$$

Once the search list is empty, the algorithm terminates and the current $\overline{F}$ and its corresponding solution form the global optimum. A flow chart of the presented algorithm can be found in Appendix A.

Remark 3.1: The performance of the algorithm is also dependent on the rule for selecting a sub-problem from the search list. This can be done, for example, by applying depth-first search or breadth-first search. In their book, Domschke and Drexl [9] give a brief overview of different approaches.

Let us now give a small example of Dakin's branch and bound algorithm.

Example 3.1: (*Branch & Bound algorithm*)
Consider the optimization problem

$$\begin{aligned} &\min_{\mathbf{x}} \ 200x_1 + 500x_2 \\ \text{s.t.} \quad & x_1 + 2x_2 \geq 2 \\ & 2x_2 \geq 1 \\ & x_1, x_2 \in [0,1] \end{aligned}$$

By having a closer look at the problem, we can find a feasible initial solution $\mathbf{x}_0 = (1,1)^T$ with $\underline{F}_0 = 700$. Thus, we define the upper bound of the objective function value by $\overline{F} := 700$.

Problem P_0: In solving the relaxation $P_0^{'}$, we obtain $\mathbf{x}_0^* = (1, \frac{1}{2})^T$ with $\underline{F}_0 = 450$. Since $\underline{F} < \overline{F}$ but $\mathbf{x}_0^*$ is not feasible in P_0, we have to branch the initial problem into the sub-problems

$$P_1 := \{\mathbf{x}_0 \in X(P_0) \mid x_2 \leq \lfloor x_2^* \rfloor = 0\} \text{ and } P_2 := \{\mathbf{x}_0 \in X(P_0) \mid x_2 \geq \lceil x_2^* \rceil = 1\}.$$

Figure 2 shows the search tree of the optimization problem.

Problem P_1: With the additional constraint $x_2 \leq 0$, it follows that $x_2 = 0$. As such, $X(P_1^{'}) = \emptyset$ and thus, $X(P_1) = \emptyset$ as well. Therefore, we can remove P_1 from the list and continue with P_2.

Problem P_2: Here, the additional constraint $x_2 \geq 1$ leads to $x_2 = 1$. After solving the relaxation $P_2^{'}$, we obtain $\mathbf{x}_2^* = (0, 1)^T$ with $\underline{F}_2 = 500 < \overline{F}$. Since $\mathbf{x}_2^*$ is an integer solution, it is feasible in P_2 and we have thus found a new best solution. Therefore, we set $\overline{F} := 500$. Having no problem left in the search list, the algorithm terminates with the optimal solution $\mathbf{x}_2^*$.

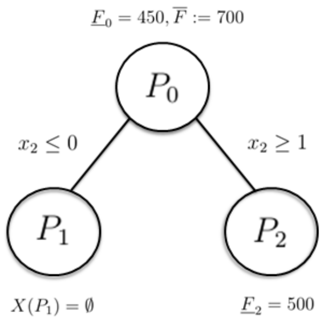


Figure 2: Search tree of the branch & bound example [9].

As we know from Chapter 2, the relaxation of the *minimum set coverage problem* is the *land allocation problem*. This optimization problem can be solved by applying a special variant of the simplex algorithm, called the dual bounded simplex, which will be presented in the following section.

3.2 The Dual Bounded Simplex Algorithm

In this section, we will explain the dual bounded simplex algorithm which is also known as the dual simplex with box constraints. In order to understand the algorithm itself, it is not necessary to define duality right away but we will do so later on in Chapter 4.1.1. After a brief introduction, we will give a detailed description of the algorithm and illustrate it by using the data of the optimization problem presented in Example 2.3. We will also briefly discuss the issue of primal and dual degeneracy. Additionally, a summary of the dual bounded simplex algorithm will be provided in Appendix B. An overview of the notation used in this chapter can be found in Table 3.

Symbol	**Explanation**
$i \in \{1, ..., n\}$	Index for decision variables
$j \in \{1, ..., m\}$	Index for constraints
$s \in \{1, ..., n+m\}$	Column index of $\hat{A}$ and $\hat{\mathbf{c}}^T$
k	Index of the pivot row
l	Index of the pivot column
$A = (a_{ji})$	Coefficient matrix of the decision variables
$\tilde{A} \in \mathbb{R}^{m \times m}$	Coefficient matrix of the slack variables
$\hat{A} = (\hat{a}_{js})$	Coefficient matrix of the decision and slack variables
$\mathbf{b} = (b_1, ..., b_j)^T$	Vector of the RHS
$\mathbf{c} = (c_1, ..., c_n)^T$	Cost vector of decision variables
$\tilde{\mathbf{c}} = (\tilde{c}_1, ..., \tilde{c}_m)^T$	Cost vector of slack variables
$\hat{\mathbf{c}} = (\hat{c}_1, ..., \hat{c}_{n+m})^T$	Cost vector of decision and slack variables
$\mathbf{x} = (x_1, ..., x_n)^T$	Vector of decision variables
$\tilde{\mathbf{x}} = (\tilde{x}_1, ..., \tilde{x}_m)^T$	Vector of slack variables of the constraints
$\hat{\mathbf{x}} = (\hat{x}_1, ..., \hat{x}_{n+m})^T$	Vector of decision and slack variables
$\mathbf{x}' = (x'_1, ..., x'_n)^T$	Vector of complementary variables of $\mathbf{x}$
$\mathbf{u} = (u_1, ..., u_n)^T$	Vector of upper bounds of the decision variables
z	Objective function value

Table 3: Notation.

Let us recall the structure of the *static minimum set coverage problem* (2.2) as

$$\min\{\mathbf{c}^T\mathbf{x} \mid A\mathbf{x} \geq \mathbf{b}, 0 \leq \mathbf{x} \leq \mathbf{u}\},$$

with $\mathbf{c} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^m_+$, $A \in \mathbb{R}^{m \times n}$ and $\mathbf{u} := \mathbb{1}$ being the vector of upper bounds of the variables x_i. Transforming this into the initial form required by the simplex algorithm we obtain

$$-\max\{-\mathbf{c}^T\mathbf{x} \mid -A\mathbf{x} \leq -\mathbf{b}, 0 \leq \mathbf{x} \leq \mathbf{u}\}. \tag{3.1}$$

Since $-\mathbf{b} \leq 0$, the problem is primal infeasible, but with $-\mathbf{c} \leq 0$, it is dual feasible instead. This means that in order to find an optimal solution to the optimization problem, we can use Lemke's dual simplex algorithm [7]. With respect to the upper bounds of the decision variables, we will use a special variant for bounded variables, called the dual bounded simplex algorithm. This algorithm was presented by Wagner in 1958 [26].

In order to use the dual bounded simplex algorithm, the matrix A must be of full rank. This means that there are no lines or columns consisting of zeros only, nor are any of them linearly dependent. If A is, for example, the matrix of the species representation levels, like in the *minimum set coverage problem*, the targets (here b_j) must not be greater than the total amount of the respective individuals available. Therefore, to avoid infeasibility, it must hold that

$$\sum_{i \in I} a_{ji} \geq b_j$$

for all $j \in J$.

Once the optimization problem has been converted to the form (3.1), we add a vector of m so-called slack variables $\tilde{\mathbf{x}}$ to the constraints such that the inequalities become equations, i.e.

$$-\max\{-\mathbf{c}^T\mathbf{x} \mid -A\mathbf{x} + I\tilde{\mathbf{x}} = -\mathbf{b}, 0 \leq \mathbf{x} \leq \mathbf{u}, \tilde{\mathbf{x}} \geq 0\}, \tag{3.2}$$

with $I \in \mathbb{R}^{m \times m}$ being the unit matrix. Furthermore, we add another vector of n slack variables $\mathbf{x}' \geq 0$ to the upper bound relations such that

$$\mathbf{x} + \mathbf{x}' = \mathbf{u}. \tag{3.3}$$

We call x'_i the complementary variable of x_i for all i in I.

Furthermore, we define the vector of the decision variables and the slack variables of the constraints by

$$\hat{\mathbf{x}} := (\mathbf{x}, \tilde{\mathbf{x}})^T \in \mathbb{R}^{n+m}$$

and the corresponding cost vector by

$$\hat{\mathbf{c}} := (-\mathbf{c}, \tilde{\mathbf{c}}) \in \mathbb{R}^{n+m}.$$

Additionally, let

$$\hat{A} := (-A, \tilde{A}) \in \mathbb{R}^{m \times n+m},$$

with $\hat{a}_{js}$ being the element of the j-th row and s-th column of matrix $\hat{A}$.

Let us now define the terms basis, basic variable and feasible basis solution which we are going to need especially in Section 3.2.2 about primal degeneracy.

Definition 3.1: (*Basis*, [7])
We call a vector B of the indices of m linearly independent columns of the matrix $[-AI]$ a basis of $\mathbb{R}^m$. All columns that are not part of B are referred to as the non-basis N.

Definition 3.2: (*Basic variable*, [7])
A variable x_j with $j \in B$ is called a basic variable. The vector of all basic variables is denoted by $\mathbf{x}_B$. All other variables are non-basic variables and are denoted by $\mathbf{x}_N$.

Definition 3.3: (*Feasible basis solution*, [7])
A solution $\mathbf{x}$ of the feasible set $\hat{X} := \{\hat{\mathbf{x}} \in \mathbb{R}^{n+m} \mid -A\mathbf{x} + I\tilde{\mathbf{x}} = -\mathbf{b}, 0 \leq \mathbf{x} \leq \mathbf{u}, \tilde{\mathbf{x}} \geq 0\}$ is called a feasible basis solution, if there exists a basis B with

(i) $[-AI]_B \mathbf{x}_B = -\mathbf{b}$,

(ii) $\mathbf{x}_N = 0$,

(iii) $x_j \geq 0 \; \forall \, j \in B$.

In this case, B is called a feasible basis.

Theorem 3.2: A point $\hat{\mathbf{x}} \in \hat{X}$ is a vertex of $\hat{X}$, if and only if $\hat{\mathbf{x}}$ is a basis solution.

The proof of this theorem can be found in Burkhard and Zimmermann [7, p. 33].

3.2.1 Algorithm

Analogous to the primal simplex, the dual simplex algorithm starts with a feasible basis, though here, the initial basis must be dual feasible. The algorithm moves along adjacent feasible vertices until it finds a dual optimal solution. Here, optimal means that the solution is dual as well as primal feasible. Even though at each step of the algorithm the current solution is primal optimal, the aim is to reach also primal feasibility, while retaining dual feasibility [7].

If a variable x_i is part of the basis and exceeds its upper bound u_i during the optimization process, the idea of the bounded simplex is that it gets substituted by its complementary variable $x'_i = u_i - x_i$. Thus, the algorithm guarantees that the variables are always within their bounds and once an optimal solution has been found, the complementary variables can be re-substituted easily. This method reduces some of the computational effort given that it is not necessary anymore to introduce additional constraints for variable boundaries. Still, the algorithm becomes a bit more complex, since more value comparisons are needed, e.g. due to the boundary checks at each pivot step [26].

Table 4 shows the initial dual simplex tableau.

	$\mathbf{x}$	$\tilde{\mathbf{x}}$	
z	$-\mathbf{c}$	$\tilde{\mathbf{c}}$	0
$\tilde{\mathbf{x}}$	$-A$	I	$-\mathbf{b}$

Table 4: Initial dual simplex tableau.

The first line of the tableau holds the names of the decision variables and the slack variables of the constraints. Below these, in the first column, we note down a z for the objective function value, which is equal to 0 at the beginning of the optimization process. Therefore, we write 0 in the last column. The second till the $n+m$-th column of the second line contain the costs of each variable, whereupon we use the negative cost vector $-\mathbf{c}$ for the decision variables and a vector of zero cost, i.e. $\tilde{\mathbf{c}} = \mathbf{0}$ for the slack variables.

The first column of the following lines consists of the names of the basis variables. Initially, these are all the slack variables of the constraints, since we start at the dual feasible basis $\tilde{\mathbf{x}}_{initial} := -\mathbf{b}$. All non-basis variables are considered to be zero. From the second column on, we note down the negative of matrix A followed by the coefficient matrix of the slack variables $\tilde{A}$ which initially equals the unit matrix. In the last column, we add the negative of the RHS $\mathbf{b}$.

Before we start the optimization itself, we check if the current solution is already optimal, i.e. $-b_j \geq 0$ for all j, $\hat{c}_s \leq 0$ for all $s \in \{1, ..., n+m\}$ and all variables are within their bounds. If this is not the case, we choose a pivot row by selecting the most negative variable, i.e. the row with the smallest negative b_j. Let k be the index of the pivot row. Then, the corresponding pivot element $\hat{a}_{kl}$ is the one that fulfils

$$\min\left\{\frac{z - \hat{c}_s}{-\hat{a}_{ks}}\right\} \forall s \in \{1, ..., n+m\}, \hat{a}_{ks} < 0,$$

with l being the index of the pivot column. Hence, the k-th variable is leaving the basis and will be replaced by the l-th variable. If $\hat{a}_{ks} \geq 0$ for all s, then, the given optimization problem has no feasible solution.

Now, we set up a tentative new tableau by carrying out the usual pivot step. This means that we divide all elements of the pivot row by the pivot element, i.e.

$$\hat{a}'_{ks} := \frac{\hat{a}_{ks}}{\hat{a}_{kl}} \text{ and } -b'_k := \frac{-b_k}{\hat{a}_{kl}}$$

so that $\hat{a}'_{kl} = 1$. Leaving out the pivot element itself, we generate zeros in the pivot column by multiplying the pivot row with the factor $-\hat{a}_{jl}$ and adding it to the j-th constraint, i.e

$$\hat{a}'_{js} := \hat{a}_{js} + \hat{a}'_{ks} \cdot (-\hat{a}_{jl}) \text{ and } -b'_j := -b_j + (-b'_k) \cdot (-\hat{a}_{jl}) \; \forall j, j \neq k.$$

Having a look at the values of the basis variables in the last column, we can check if they are within their bounds. Note that due to (3.3), the complementary variables are bounded as well. Only the decision variables and their complements can exceed their upper bounds, since the slack variables are unbounded above. So in case a variable, say x_p, in the r-th row of the basis is above its upper bound, we substitute it by its complementary variable x'_p.

In doing so, we first change the sign of all elements of row r in the simplex tableau, leaving out the factor 1 which appears in the correspondent column of x_p. Then, we replace the label and the value of x_p by its complementary variable, i.e.

$$x'_p = u_p - x_p$$

is going to be the new RHS of the correspondent constraint.

Certainly, the substitution also affects the objective function. There, we add the product $c_p u_p$ to the current objective function value z, i.e.

$$z' = z + c_p u_p$$

and change the sign of c_p.

Remark 3.2: In order to explain what is happening at the substitution of a variable, consider the optimization problem after the pivot step

$$-\max\left\{\hat{\mathbf{c}}'^T\hat{\mathbf{x}} \mid \hat{A}'\hat{\mathbf{x}} = -\mathbf{b}, \hat{\mathbf{x}} = (\mathbf{x}, \tilde{\mathbf{x}})^T, 0 \leq \mathbf{x} \leq \mathbf{u}, \tilde{\mathbf{x}} \geq 0\right\}.$$

In more detail, the constraints can be written as

$$\begin{array}{cc} & \begin{array}{ccccccccc} \phantom{\hat{a}'_{1,1}x_1} & & p & & & & & & \end{array} \\ r & \begin{pmatrix} \hat{a}'_{1,1}x_1 & +\ldots & +0x_p & +\ldots & +\hat{a}'_{1,n}x_n & +\hat{a}'_{1,n+1}\tilde{x}_1 & +\ldots & +\hat{a}'_{1,n+m}\tilde{x}_m \\ \vdots & & \vdots & & \vdots & \vdots & & \vdots \\ \hat{a}'_{r,1}x_1 & +\ldots & +1x_p & +\ldots & +\hat{a}'_{r,n}x_n & +\hat{a}'_{r,n+1}\tilde{x}_1 & +\ldots & +\hat{a}'_{r,n+m}\tilde{x}_m \\ \vdots & & \vdots & & \vdots & \vdots & & \vdots \\ \hat{a}'_{m,1}x_1 & +\ldots & +0x_p & +\ldots & +\hat{a}'_{m,n}x_n & +\hat{a}'_{m,n+1}\tilde{x}_1 & +\ldots & +\hat{a}'_{m,n+m}\tilde{x}_m \end{pmatrix} \end{array} = \begin{pmatrix} -b'_1 \\ \vdots \\ -b'_r \\ \vdots \\ -b'_m \end{pmatrix}.$$

Let us now substitute x_p by its complement using $u_p - x'_p$. Thus, we obtain

$$\begin{array}{cc} & \begin{array}{ccccc} \phantom{\hat{a}'_{1,1}x_1} & & p & & \end{array} \\ r & \begin{pmatrix} \hat{a}'_{1,1}x_1 & +\ldots & +0(u_p - x'_p) & +\ldots & +\hat{a}'_{1,n+m}\tilde{x}_m \\ \vdots & & \vdots & & \vdots \\ \hat{a}'_{r,1}x_1 & +\ldots & +1(u_p - x'_p) & +\ldots & +\hat{a}'_{r,n+m}\tilde{x}_m \\ \vdots & & \vdots & & \vdots \\ \hat{a}'_{m,1}x_1 & +\ldots & +0(u_p - x'_p) & +\ldots & +\hat{a}'_{m,n+m}\tilde{x}_m \end{pmatrix} \end{array} = \begin{pmatrix} -b'_1 \\ \vdots \\ -b'_r \\ \vdots \\ -b'_m \end{pmatrix}.$$

Substracting $(0, \ldots, u_p, \ldots, 0)^T$ and multiplying the r-th row by -1, gives

$$\begin{array}{cc} & \begin{array}{ccccc} \phantom{-\hat{a}'_{r,1}x_1} & & p & & \end{array} \\ r & \begin{pmatrix} \hat{a}'_{1,1}x_1 & +\ldots & +0x'_p & +\ldots & +\hat{a}'_{1,n+m}\tilde{x}_m \\ \vdots & & \vdots & & \vdots \\ -\hat{a}'_{r,1}x_1 & -\ldots & +1x'_p & -\ldots & -\hat{a}'_{r,n+m}\tilde{x}_m \\ \vdots & & \vdots & & \vdots \\ \hat{a}'_{m,1}x_1 & +\ldots & +0x'_p & +\ldots & +\hat{a}'_{m,n+m}\tilde{x}_m \end{pmatrix} \end{array} = \begin{pmatrix} -b'_1 \\ \vdots \\ u_p + b'_r \\ \vdots \\ -b'_m \end{pmatrix}.$$

Focusing on the objective function, the substitution of x_p leads to

$$\begin{array}{rll} & z & = -c_1x_1 - \ldots - c_px_p - \ldots - c_nx_n + \tilde{c}_1\tilde{x}_1 + \ldots + \tilde{c}_m\tilde{x}_m \\ & & = -c_1x_1 - \ldots - c_p(u_p - x'_p) - \ldots + \tilde{c}_m\tilde{x}_m \\ \Leftrightarrow & z + c_pu_p & = -c_1x_1 - \ldots + c_px'_p - \ldots + \tilde{c}_m\tilde{x}_m \end{array},$$

which explains the transformation of the simplex tableau stated above.

After the substitution, we obtained the finalized new tableau. Taking this, we go back to the optimality test and start the next iteration of the algorithm, until an optimal solution has been found. The same applies, if no basis variable exceeds its upper bound.

Once the algorithm has terminated, an optimal solution can be read from the final tableau. All non-basis variables take the value 0, whereas the values of the basis variables can be found in the last column of the tableau. Note that all complementary variables must be re-substituted by

$$x_i = u_i - x_i'.$$

In some special situations it may happen that there are multiple optimal solutions or that a solution is degenerate. These cases will be covered in Section 3.2.2.

If at any step of the algorithm all elements of the RHS $-\mathbf{b}$ are positive (the problem is primal feasible), but there is at least one positive element in the cost vector $\hat{\mathbf{c}}$, the problem is not dual feasible anymore and we have to continue the optimization using the primal simplex algorithm. Thus, it may happen that during the optimization, problem solvers switch between the primal and the dual simplex algorithm.

Let us now give an example of the dual bounded simplex algorithm.

Example 3.2: (*Dual bounded simplex algorithm*)
Consider the linear optimization problem from example 2.3. Transforming it to the required form by the simplex algorithm and adding slack variables, we obtain

$$\begin{array}{lrrrrrcr} -\max\limits_{\mathbf{x}} & -400x_1 - 300x_2 - 600x_3 \\ \\ \text{s.t.} & -3x_1 & & -4x_3 & +1\tilde{x}_1 & & = & -3 \\ & -6x_1 & -4x_2 & -3x_3 & & +1\tilde{x}_2 & = & -7 \end{array}$$

for the objective function and the constraints, and

$$\begin{array}{rcl} x_1 + x_1' & = & 1 \\ x_2 + x_2' & = & 1 \\ x_3 + x_3' & = & 1 \end{array}$$

for the upper bound relations. Now, we set up the initial dual simplex tableau

	x_1	x_2	x_3	$\tilde{x}_1$	$\tilde{x}_2$	
z	-400	-300	-600	0	0	0
$\tilde{x}_1$	-3	0	-4	1	0	-3
$\tilde{x}_2$	-6	-4	-3	0	1	-7

Table 5: Initial dual simplex tableau of the illustrative data set.

This tableau is dual feasible, but primal infeasible, since the values of the RHS are negative. Therefore, the tableau is not optimal and we choose the pivot row by selecting the most negative b_j, i.e. $b_2 = -7$. We obtain the pivot element by determining the element $\hat{a}_{2s} \leq 0$, which corresponds to

$$\min\left\{\frac{0+400}{6}, \frac{0+300}{4}, \frac{0+600}{3}\right\} = 66\tfrac{2}{3}.$$

The result is $\hat{a}_{21} = -6$. This implies that $\tilde{x}_2$ is going to leave, and x_1 is going to enter the basis.

Carrying out the pivot step leads to the tentative new tableau

	x_1	x_2	x_3	$\tilde{x}_1$	$\tilde{x}_2$	
z	0	$-\frac{100}{3}$	-400	0	$-\frac{200}{3}$	$466\frac{2}{3}$
$\tilde{x}_1$	0	2	$-\frac{5}{2}$	1	$-\frac{1}{2}$	$\frac{1}{2}$
x_1	1	$\frac{2}{3}$	$\frac{1}{2}$	0	$-\frac{1}{6}$	$\frac{7}{6}$

Table 6: Tentative new tableau.

In the next step, we check if all basis variables are within their bounds. For $\tilde{x}_1 = \frac{1}{2}$ this is the case, but $x_1 = \frac{7}{6}$ exceeds its upper bound by $\frac{1}{6}$. Therefore, we have to substitute this variable by its complement

$$x_1' = 1 - x_1 = -\tfrac{1}{6}$$

which leads to the following final new tableau

	x_1'	x_2	x_3	$\tilde{x}_1$	$\tilde{x}_2$	
z	0	$-\frac{100}{3}$	-400	0	$-\frac{200}{3}$	$466\frac{2}{3}$
$\tilde{x}_1$	0	2	$-\frac{5}{2}$	1	$-\frac{1}{2}$	$\frac{1}{2}$
x_1'	1	$-\frac{2}{3}$	$-\frac{1}{2}$	0	$\frac{1}{6}$	$-\frac{1}{6}$

Table 7: Finalized new tableau.

Here, we changed all signs of the second row, leaving out the factor $\hat{a}_{21} = 1$. Additionally, we changed the label of x_1 to x'_1 and entered the value of the complementary variable into the RHS. Since the cost coefficient of x_1 is 0, nothing happened in the objective function.

Now, we start the next iteration of the algorithm. With $b_2 \leq 0$, the tableau is not optimal and just like before, we continue by choosing the second row as the pivot row. Furthermore, we identify $\hat{a}_{22} = -\frac{2}{3}$ as the pivot element, since it fulfils

$$\min\left\{\frac{466\frac{2}{3} + \frac{100}{3}}{\frac{2}{3}}, \frac{466\frac{2}{3} + 400}{\frac{1}{2}}\right\} = 750.$$

With x'_1 leaving, and x_2 entering the basis, we carry out the next pivot step and obtain

	x'_1	x_2	x_3	$\tilde{x}_1$	$\tilde{x}_2$	
z	-50	0	-375	0	-75	475
$\tilde{x}_1$	3	0	-4	1	0	0
x_2	$-\frac{3}{2}$	1	$\frac{3}{4}$	0	$-\frac{1}{4}$	$\frac{1}{4}$

Table 8: Optimal tableau.

As it can be seen in the tableau, all elements of $\hat{\mathbf{c}}$ are non-positive and all elements of the RHS are positive. In addition, all variables are within there bounds. Hence, the tableau is optimal and we can read off the solution

$$(x'_1, x_2, x_3, \tilde{x}_1, \tilde{x}_2)^T = (0, \tfrac{1}{4}, 0, 0, 0)^T,$$

where all non-basis variables take the value 0 and all basis variables the values of the corresponding RHS.

By re-substituting x'_1 by $x_1 = 1 - x'_1 = 1$, we obtain the optimal solution

$$\mathbf{x}^* = (1, \tfrac{1}{4}, 0)^T,$$

with $z^* = 475$.

There are some special situations which can appear during the optimization of a linear problem. The following section will cover the cases of primal and dual degeneracy, illustrating each with a small example.

3.2.2 Primal & Dual Degeneracy

Definition 3.4: (*Primal degeneracy*, [9])
If there exists at least one $j \in B$ with $x_j = 0$, the basic solution is called primal degenerate.

Primal degeneracy implies that there exist several basis solutions for the same optimal vertex. This occurs, if an optimization problem is over-determined, i.e. there are more constraints than variables. Graphically, we can choose among several constraint combinations in order to describe the same optimal solution.

If we substitute a variable x_i by its complement $x_i^{'}$ for at least one $i \in I$ during the optimization process, we also change the set $\hat{X}$ to $\hat{X}'$. In such a case, a basis solution $\hat{\mathbf{x}}$ is not a vertex of $\hat{X}$, but of $\hat{X}'$. In order for it to become a vertex of $\hat{X}$, the respective variables must be re-substituted.

According to the following scheme, we can thus decide, whether a vertex of the initial optimization problem is degenerate or not:

(i) If $0 < x_i \leq 1$ and $0 < x_i^{'} < 1$ for all $i \in I \cap B$, then the basis solution and the vertex are non-degenerate.

(ii) If there exists an $i \in B$ with $\hat{x}_i = 0$, then the basis solution and the vertex are degenerate.

(iii) If there exists an $i \in I \cap B$ with $x_i^{'} = 1$, then the basis solution is non-degenerate, but the vertex is degenerate.

(iv) If there exists an $i \in I \cap B$ with $x_i^{'} = 0$, then the basis solution is degenerate, but the vertex is non-degenerate.

Example 3.3: (*Primal degeneracy*)
If we have again a look at the final optimal simplex tableau in Table 8, we can see that the optimal solution $\mathbf{x}^* = (1, \frac{1}{4}, 0)^T$ is degenerate, since the basis variable $\tilde{x}_1$ takes the value 0. If we choose the first row as the new pivot row and execute another pivot with the pivot element $\hat{a}_{13} = -4$, we obtain the following new optimal tableau

	x'_1	x_2	x_3	$\tilde{x}_1$	$\tilde{x}_2$	
z	$-331\frac{1}{4}$	0	0	$-93\frac{3}{4}$	-75	475
x_3	$-\frac{3}{4}$	0	1	$-\frac{1}{4}$	0	0
x_2	$-\frac{15}{16}$	1	0	$-\frac{3}{16}$	$-\frac{1}{4}$	$\frac{1}{4}$

Table 9: New optimal tableau.

Obviously, this tableau shows the same optimal solution as before, but the basis has changed from former $(\tilde{x}_1, x_2)$ to (x_3, x_2) now. Also, in the plot of this optimization problem in Figure 1, it can be seen that the point P_2, which corresponds to our optimal solution, is degenerate: we are considering a three-dimensional problem, which means that there are only 3 constraints needed in order to define the optimal vertex. However, P_2 is at the intersection of 4 constraints - the light and dark blue (target restrictions), and the planes of the two variable restrictions $x_1 \leq 1$ and $x_3 \geq 0$.

Thus, there are $\binom{4}{3} = 4$ possibilities to describe $\mathbf{x}^*$, i.e. 4 different basic solutions for one optimal solution.

Primal degeneracy can cause the simplex algorithm to cycle through the basic solutions of a vertex, although there are ways of modifying the algorithm in order to avoid such situations [9]. In this thesis, we will focus more on the effects of degeneracy on shadow prices. We will have a closer look into the matter in Chapter 4.1.3.

Definition 3.5: (*Dual degeneracy*, [9])
If at least one of the non-basis variables has zero cost in an optimal tableau, the optimization problem is called dual degenerate.

Analogously to primal degeneracy, dual degeneracy occurs if at least one of the basis variables of the corresponding dual problem takes the value 0 [9]. In case of dual degeneracy, an optimization problem has multiple optimal solutions. These can be obtained by executing another pivot, where the entering variable is one of the non-basis variables with zero cost.

Corollary 3.1: (*Parametric solution*, [9])
Let $\mathbf{x}_1^*$ and $\mathbf{x}_2^*$ be two optimal basic solutions of a dual degenerate optimization problem. Then, all non-basic solutions which can be obtained by the

convex combination

$$\mathbf{x}^* = \lambda \cdot \mathbf{x}_1^* + (1-\lambda) \cdot \mathbf{x}_2^*, \;\; 0 \le \lambda \le 1$$

are optimal as well.

The corollary implies that in case of dual degeneracy, all optimal solutions lie on a line between two optimal basic solutions. Graphically, this means that the objective function is parallel to a non-redundant constraint. The following example will illustrate this correlation.

Example 3.4: (*Dual degeneracy*)
Consider the optimization problem

$$-\max_{\mathbf{x}} \; -200x_1 - 400x_2$$

$$\text{s.t.} \quad -2x_1 - 4x_2 + 1\tilde{x}_1 = -1$$

with the upper bound relations

$$\begin{array}{rcl} x_1 + x_1' &=& 1 \\ x_2 + x_2' &=& 1 \end{array} \; .$$

This leads to the following initial dual simplex tableau

	x_1	x_2	$\tilde{x}_1$	
z	-200	-400	0	0
$\tilde{x}_1$	-2	-4	1	-1

Table 10: Initial tableau.

Since the ratios for identifying the pivot column are equal, we pick $\hat{a}_{11} = -2$ as the pivot element. After executing a pivot step, we obtain

	x_1	x_2	$\tilde{x}_1$	
z	0	0	-100	100
x_1	1	2	$-\frac{1}{2}$	$\frac{1}{2}$

Table 11: Optimal tableau 1.

This tableau is optimal and we can read off the solution $\mathbf{x}_1^* = (\frac{1}{2}, 0)^T$. Having a closer look at the cost line of the tableau, it can be seen that the non-basis variable x_2 has zero cost, i.e. $\hat{c}_2 = 0$. Therefore, the optimization problem is dual degenerate and we can obtain another optimal solution by letting x_2 enter the basis and executing another pivot step:

	x_1	x_2	$\tilde{x}_1$	
z	0	0	-100	100
x_2	$\frac{1}{2}$	1	$-\frac{1}{4}$	$\frac{1}{4}$

Table 12: Optimal tableau 2.

This tableau gives the optimal solution $\mathbf{x}_2^* = (0, \frac{1}{4})^T$. In Figure 3, one can see that the target constraint is parallel to the objective function (red). Thus, the simplex algorithm detected both vertices, $P_1 = \mathbf{x}_1^*$ and $P_2 = \mathbf{x}_2^*$, as optimal solutions. Furthermore, all points on the line between these solutions are optimal, which means that we can calculate a parametric optimal solution by applying the result of Corollary 3.1:

$$\mathbf{x}^* = \lambda \cdot (\tfrac{1}{2}, 0)^T + (1 - \lambda) \cdot (0, \tfrac{1}{4})^T, \quad 0 \leq \lambda \leq 1.$$

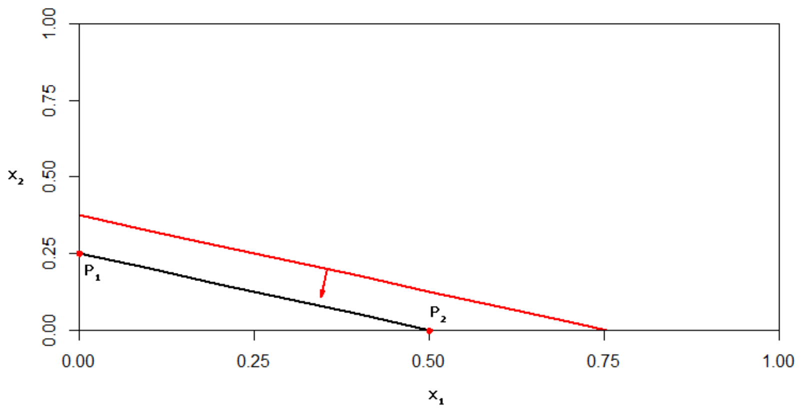


Figure 3: Plot of a dual degenerate optimization problem. All points on the line between P_1 and P_2 are optimal.

With the branch-and-bound and the dual bounded simplex algorithm, we already presented two direct resolution methods. In the section below, we are now going to discuss the heuristic approach of simulated annealing.

3.3 Simulated Annealing (Marxan)

Simulated annealing is an algorithm for solving a diversity of optimization problems which mimics the physical process of heating metal and cooling it again. Depending on the temperature control, this treatment alters the physical and chemical properties of the metal in order to make it more workable. At the beginning of the process, the temperature is set very high and the molecules whirl around, sometimes improving and sometimes degrading the structure of the metal. With a decreasing temperature, the molecules tend to arrange themselves in proper order which means that the acceptance of degrading movements becomes lower until the process stops [25].

In the first part of this section, we will give a brief explanation of the algorithm itself, before focusing on its application to conservation prioritization problems in the second part. There, we will introduce the software Marxan which uses simulated annealing as an underlying algorithm in order to solve the *minimum set coverage problem*, presented in Chapter 2.

3.3.1 Algorithm

In Chapter 5.3.1 of [18], Moilanen and Ball discuss the topic of simulated annealing in spatial optimization. In doing so, they are describing the simulated annealing algorithm as follows:

Let $\mathbf{x}_0$ be an initial feasible solution which has been generated either randomly or by some heuristic algorithm and let T_0 be the initial temperature. Now, we calculate new potential solutions iteratively. For $r = 1, ..., R$, set the temperature parameter

$$T := T_0 f(r),$$

where $f(r)$ is a decreasing function of iteration round r. Then, generate a new solution candidate $\mathbf{x}_0'$ as a stochastic modification of $\mathbf{x}_0$.

If $\mathbf{x}_0'$ improves the system, i.e.

$$\Delta z := z(\mathbf{x_0'}) - z(\mathbf{x_0}) < 0$$

(when minimizing), the new solution will always be accepted and we set

$$x_0 := x_0'$$

for the next iteration. If x_0' is performing worse than the initial solution, it will only be accepted with probability

$$exp\left(\frac{-\Delta z}{T}\right).$$

One can see, that the algorithm also accepts non-improving solutions in order to escape local optima. At the beginning of the optimization process, this happens very often but the likelihood diminishes with every further iteration. Once all R iterations have been made, or after a certain number of iterations without finding any improving solution, the algorithm terminates.

Figure 4 illustrates the optimization process of the simulated annealing algorithm. The objective is to find the global minimum of function $z(x)$. The algorithm starts at the initial solution x_0 (the first black ball on the left) and then moves to another one with a better performance of the objective function (second black ball). At this point, if the algorithm only accepted improving solution candidates, it would get trapped in a local optimum. However, in the example, the temperature is still high enough so that it can escape this solution by accepting another, which is performing less well than the current solution (i.e. the first green ball). After two more iterations choosing only improving solutions, the algorithm finally finds the global optimum x^* (red ball) and terminates.

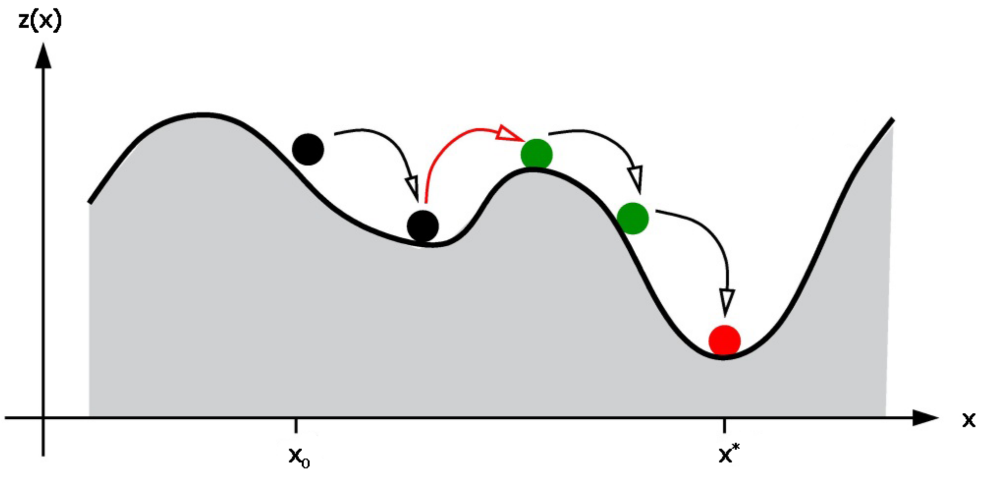


Figure 4: Example of the simulated annealing algorithm [25].

Simulated annealing is a heuristic algorithm which means that it does not necessarily find a global optimum, even though the likelihood increases with the number of iterations. Thus, a given result should mostly be considered as an approximate solution to the optimization problem. Furthermore, the efficiency of the algorithm implementation is highly dependent on the way of how new solutions are generated [18].

The following pseudo-code gives a brief summary of the simulated annealing algorithm presented above.

Algorithm 3.1: (*Pseudo-code of the simulated annealing algorithm*, [18])

```
0. Generate initial feasible solution $\mathbf{x}_0$. Let $z(\mathbf{x})$ be the performance of
   solution candidate $\mathbf{x}_0$.
1. FOR $r = 1, 2, ..., R$ iterations DO
   1.1 SET temperature $T = T_0 f(r)$, where $T_0$ is initial temperature and
       $f(r)$ a decreasing function of iteration round $r$.
   1.2 Generate next solution candidate $\mathbf{x}_0'$ as stochastic modification of
       current solution $\mathbf{x}_0$.
   1.3 SET $\Delta z := z(\mathbf{x_0'}) - z(\mathbf{x_0})$.
   1.4 IF $\Delta z < 0$ (when minimizing), accept improving solution and set
       $\mathbf{x}_0 := \mathbf{x}_0'$.
       ELSE accept $\mathbf{x}_0'$ with probability $exp\left(\frac{-\Delta z}{T}\right)$.
```

3.3.2 Marxan

Marxan is a freely available software which uses simulated annealing in order to solve the type of spatial conservation prioritization problems which has been introduced as the *minimum set coverage problem* in Chapter 2. The name Marxan stands for 'marine reserve design using spatially explicit annealing'. The software has already been used by over 1,200 organizations from more than 100 countries and one of the biggest projects where it has been applied, was the rezoning of the Great Barrier Reef in 2004. [18, 21]

Marxan "interacts with a variety of geographical information system (GIS) tools [and it] designs clumped reserve systems that make sense to a policy maker or planner" [18]. Mathematically, it solves optimization problems of the form

$$\begin{aligned} \min_{\mathbf{x}\in\{0,1\}^n} \quad & \sum_{i\in I} c_i x_i + b \sum_{i\in I}\sum_{h\in I} x_i(1-x_h)cv_{ih} & & \\ \text{s.t.} \quad & \sum_{i\in I} r_{ji} x_i \geq t_j & \forall\, j \in J & \qquad (3.4) \\ & x_i \in \{0,1\} & \forall\, i \in I, & \end{aligned}$$

using the notation of Chapter 2 and with cv_{ih} being elements of the connectivity matrix CV. This matrix contains the cost of the connection of sites i and h, though cv_{ih} could also be set as the length of their physical boundary. The parameter b is the so-called boundary multiplier and serves as a weight of how important it is to have a highly connected reserve network [18]. If, for example, a conservation planner prefers few big reserve systems to many small ones, the boundary multiplier must be set with a high value.

Thus, Marxan aims to minimize a combination of the cost of the reserve system $\sum_{i\in I} c_i x_i$ and its boundary length $b\sum_{i\in I}\sum_{h\in I} x_i(1-x_h)cv_{ih}$, whilst meeting the target t_j of each considered feature. Note that if we do not take into account boundaries, i.e. $b := 0$, we obtain the *minimum set coverage problem* (2.3).

After investigating several different methods, the developers of Marxan found that simulated annealing would be the best to solve the above presented optimization problem. It finds good solutions quickly for problems of different sizes. Furthermore, simulated annealing makes the software very flexible towards changes in the type of a problem, since non-linearities and other complexities can be added easily [18].

Marxan generates multiple near-optimal solutions which is not necessarily a drawback, considering that "for practical and political reasons, finding the single best solution is not that useful in conservation planning" [18]. In fact, decision makers prefer having a set of alternative solutions, seeing that it may not be possible to put into practice a single optimal solution.

The following small example gives an illustration of how Marxan solves a *minimum set coverage problem.*

Example 3.5: (*Marxan optimization process*, [24])
Consider the optimization problem (2.17) from Chapter 2. Since we are not taking into account any boundaries, the boundary multiplier is set 0. In solving the problem, Marxan starts with a randomly chosen feasible solution, say $\mathbf{x}_0 = (1, 0, 1)^T$. This means that the software selects sites 1 and 3 for conservation, as shown in Figure 5. This solution implies total cost of \$1,000.

Now, the algorithm selects a planning unit at random, e.g. site 2, and changes its protection status. The new objective function value of \$1,300 would not improve the solution, but with the temperature still being high enough, Marxan accepts the solution. However, the likelihood for this to happen decreases with every iteration. In the next step, the software selects site 3 and changes its status from protected to unprotected. With total cost of \$700, the new solution $\mathbf{x}_0' = (1, 1, 0)^T$ improves the current and is thus being accepted. In our little example, the algorithm terminates at this stage, having found the optimal solution.

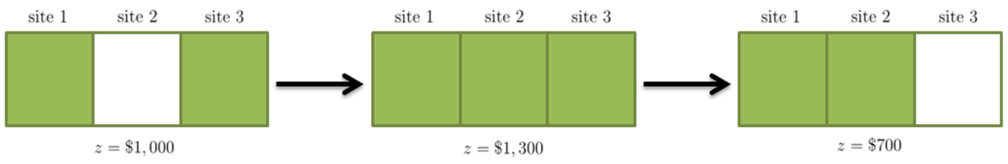


Figure 5: Marxan example using the illustrative *minimum set coverage problem* (2.17).

Now that we know how the *minimum set coverage problem* and its relaxation can be solved, we provided a basis for their sensitivity analysis which we are going to discuss in detail in the next chapter.

4 Sensitivity Analysis

In the context of conservation planning, the question of how the target setting of a feature influences total cost of the reserve system can be answered with the help of mathematical sensitivity analysis. In this chapter, we are going to focus on the sensitivity analysis based on perturbations in the right-hand side of an optimization problem. We will examine both, linear and integer linear programming problems with a main emphasis on LPs.

In the LP section, we are going to define the so-called shadow prices, the optimal solutions of the dual optimization problem, which Koopmans interprets "as guides for the coordination of allocative decisions" [17, p. 215]. Against this background, we will furthermore cover the topic of degeneracy and in the last section of this chapter, an overview of methods of sensitivity analysis for ILPs will be given.

4.1 Linear Optimization Problems

In this section, we will first give some basics about linear optimization problems, before we define different types of shadow prices and interpret them in the context of conservation planning. We start with the definitions of the canonical forms of a linear optimization problem (LP) and its corresponding dual (DLP).

Definition 4.1: *(Primal linear optimization problem* [1])
Let $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{c} \in \mathbb{R}^n_+$. Furthermore, let $A \in \mathbb{R}^{m,n}$ and $\mathbf{b} \in \mathbb{R}^m$. The primal linear optimization problem (P) in canonical form is

$$\begin{aligned} (P) \quad & \min_{\mathbf{x}} \mathbf{c}^T\mathbf{x} \\ & \text{s.t.} \quad A\mathbf{x} \geq \mathbf{b} \\ & \qquad\quad \mathbf{x} \geq 0. \end{aligned} \tag{4.1}$$

Additionally, we define the set of primal feasible solutions by

$$X(\mathbf{b}) := \{\mathbf{x} \in \mathbb{R}^n \mid A\mathbf{x} \geq \mathbf{b}, \mathbf{x} \geq 0\} \tag{4.2}$$

and the primal optimal objective function value as a function of the problem's right-hand side $\mathbf{b}$

$$F(\mathbf{b}) := \min \left\{ \mathbf{c}^T \mathbf{x} \,|\, \mathbf{x} \in X(\mathbf{b}) \right\}. \tag{4.3}$$

According to this, the set of primal optimal solutions can be defined as

$$X^*(\mathbf{b}) := \{\mathbf{x} \in X(\mathbf{b}) \,|\, \mathbf{c}^T \mathbf{x} = F(\mathbf{b})\}.$$

Definition 4.2: *(Dual linear optimization problem* [1])
Consider the primal linear optimization problem (P). The corresponding dual problem (D) can be written as

$$\begin{aligned} (D) \quad & \max_{\mathbf{y}} \ \mathbf{b}^T \mathbf{y} \\ & \text{s.t. } A^T \mathbf{y} \leq \mathbf{c} \\ & \qquad\quad \mathbf{y} \geq 0, \end{aligned} \tag{4.4}$$

with the dual variables $\mathbf{y} \in \mathbb{R}^m$.

From this formulation, we can deduce the feasible set of dual solutions

$$Y := \{\mathbf{y} \in \mathbb{R}^m \,|\, A^T \mathbf{y} \leq \mathbf{c}, \mathbf{y} \geq 0\}$$

and the dual optimal objective function value

$$G(\mathbf{b}) := \max\{\mathbf{b}^T \mathbf{y} \,|\, \mathbf{y} \in Y\}. \tag{4.5}$$

Accordingly, the set of dual optimal solutions is

$$Y^*(\mathbf{b}) := \{\mathbf{y} \in Y \,|\, \mathbf{b}^T \mathbf{y} = G(\mathbf{b})\}. \tag{4.6}$$

Remark 4.1: The dual of a dual linear optimization problem is the primal problem again.

Let us now contemplate some relationships between primal LPs and their duals which are important in the context of sensitivity analysis.

Theorem 4.1: (*Weak duality* [7])

(i) If $\mathbf{x} \in X(\mathbf{b})$ and $\mathbf{y} \in Y$, then

$$\mathbf{b}^T\mathbf{y} \leq \mathbf{c}^T\mathbf{x}. \tag{4.7}$$

(ii) Let $X(\mathbf{b}) \neq \emptyset$. The set Y is empty, if and only if $\mathbf{c}^T\mathbf{x}$ is unbounded below.

(iii) Let $Y \neq \emptyset$. The set $X(\mathbf{b})$ is empty, if and only if $\mathbf{b}^T\mathbf{y}$ is unbounded above.

The weak duality theorem states that the objective function value of a feasible solution of the primal problem forms an upper bound for the objective function value of a dual feasible solution.

Theorem 4.2: (*Strong duality* [7])
The dual linear problem has an optimal solution, if and only if its primal does. Let $\mathbf{x}^*$ be an optimal solution of the LP and $\mathbf{y}^*$ an optimal solution of its dual, then

$$\mathbf{c}^T\mathbf{x}^* = \mathbf{b}^T\mathbf{y}^*. \tag{4.8}$$

This means that the optimal objective function value of the primal problem can be equally described by multiplying its right-hand side $\mathbf{b}$ with the optimal dual solution $\mathbf{y}^*$.

We will use these relationships in the following section about shadow prices.

4.1.1 Shadow Prices

In this section, we assume that all optimal solutions are non-degenerate. The degenerate case will be covered later in Section 4.1.3. Before we get down to the shadow price definitions, we will give some characteristics of function F which we defined in equation (4.3). For this purpose, let us define its directional derivative first.

Definition 4.3: (*Directional derivative* [1])
The directional derivative of F at $\mathbf{b}$ in the direction of a vector $\mathbf{u} \in \mathbb{R}^m$ is defined by

$$D_{\mathbf{u}}F(\mathbf{b}) := \lim_{t \to 0^+} \frac{F(\mathbf{b} + t\mathbf{u}) - F(\mathbf{b})}{t}.$$

The following lemma summarizes some important properties of F.

Lemma 4.1:

(i) F is a monotonically increasing function with respect to the partial order on $\mathbb{R}^n_+$.

(ii) $F(\mathbf{b}) = G(\mathbf{b})$,

(iii) F is a piecewise linear and convex function.

(iv) If F is finite, then $D_u F(\mathbf{b})$ exists.

Proof:

(i) From the definition of F in (4.3) and from $X(\tilde{\mathbf{b}}) \supseteq X(\mathbf{b})$ for $\tilde{\mathbf{b}} \leq \mathbf{b}$ with respect to the partial order on $\mathbb{R}^n_+$, it follows that $F(\tilde{\mathbf{b}}) \leq F(\mathbf{b})$.

(ii) Due to Theorem 4.2, it holds that $F(\mathbf{b}) = \min\left\{\mathbf{c}^T\mathbf{x} \,|\, \mathbf{x} \in X(\mathbf{b})\right\} = \max\{\mathbf{b}^T\mathbf{y} \,|\, A^T\mathbf{y} \leq \mathbf{c}, \mathbf{y} \geq 0\} = G(\mathbf{b})$.

(iii) This characteristic of F is mentioned in [5, p. 213]. However, we will give the idea of the proof shortly in Remark 4.2.

(iv) With F being convex, we can infer the assertion from [3, p. 84].

□

Remark 4.2: The idea of showing that F is piecewise linear, is to divide the polyhedron P (created by the constraints) into a set of disjoint sub-polyhedra. If F is linear on each sub-polyhedron, then F is piecewise linear on P. Furthermore, we assume that optimal solutions are always in the vertices of P.

Let $\mathbf{y}_i \in \mathbb{R}^m$ be such that $G(\mathbf{b}) = \mathbf{b}^T\mathbf{y}_i$ for a fixed $\mathbf{b} \in \mathbb{R}^m$. Then, $\mathbf{y}_i$ is optimal for a certain set $M_i \subseteq \mathbb{R}^m$, with $\mathbf{b} \in M_i$. This means, if $\mathbf{b}$ is not in M_i, the optimal solution skips to another vertex.

Graphically, this can be seen in Figure 6. The black lines form the polyhedron and the continuous red line is the maximizing objective function for which $\mathbf{y}_1$ is the optimal solution. If we now change $\mathbf{b}$, the objective function changes as well (dashed red line). At some point, $\mathbf{y}_2$ might then become the optimal solution.

One can show that the different M_i are polyhedra themselves. Since for $\mathbf{b} \in M_i$ $G(\mathbf{b}) = \mathbf{b}^T\mathbf{y}_i$ is linear on M_i, G (and thus F) is piecewise linear on P.

Additionally, $G(\mathbf{b}) = \mathbf{b}^T\mathbf{y}_i$ is convex for all i. The maximum of convex function is convex as well and therefore, F is a convex piecewise linear function.

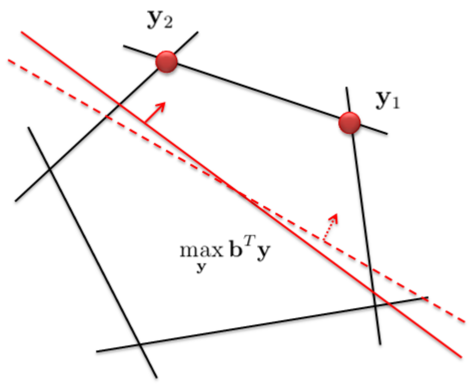


Figure 6: Example for changing optimal vertices caused by different values of $\mathbf{b}$.

Corollary 4.1: Since $F(\mathbf{b}) = G(\mathbf{b})$, we can reformulate equation (4.6) as

$$Y^*(\mathbf{b}) := \{\mathbf{y} \in Y \,|\, \mathbf{b}^T\mathbf{y} = F(\mathbf{b})\}.$$

Another characteristic of F is that its subdifferential correlates with the dual optimal objective function value.

Definition 4.4: (*Subdifferential of a convex function*)
Let F be a convex function. Then,

$$\partial F(\mathbf{b}) := \left\{\mathbf{y} \in \mathbb{R}^m \mid F(\mathbf{b}+\mathbf{u}) \geq F(\mathbf{b}) + \mathbf{u}^T\mathbf{y} \;\; \forall\, \mathbf{u} \in \mathbb{R}^m\right\}$$

is called the subdifferential of F at $\mathbf{b}$. The elements of $\partial F(\mathbf{b})$ are called subgradients.

Lemma 4.2: Let F be the function defined in (4.3). If F is finite, then,

$$\partial F(\mathbf{b}) = Y^*(\mathbf{b}).$$

In order to prove this lemma, we have to consider the following formulations of functions F and G which are equivalent to the respective definitions given in Equations (4.3) and (4.5).

As in Chapter 3.2, we add slack variables $\tilde{\mathbf{x}} \geq 0$ to the constraints of the primal optimization problem and obtain

$$F(\mathbf{b}) = \min\{\hat{\mathbf{c}}^T\hat{\mathbf{x}} \mid \hat{A}\hat{\mathbf{x}} = \mathbf{b}, \hat{\mathbf{x}} \geq 0\},$$

with $\hat{A} = (A, -I) \in \mathbb{R}^{m\times n+m}, I$ unit matrix, $\hat{\mathbf{c}} = (\mathbf{c}, \mathbf{0}) \in \mathbb{R}_+^{n+m}$ and $\hat{\mathbf{x}} = (\mathbf{x}, \tilde{\mathbf{x}})^T \in \mathbb{R}^{n+m}$. According to this formulation, the set of primal feasible solutions changes to

$$\hat{X}(\mathbf{b}) = \{\hat{\mathbf{x}} \in \mathbb{R}^{n+m} \mid \hat{A}\hat{\mathbf{x}} = \mathbf{b}, \hat{\mathbf{x}} \geq 0\}.$$

The corresponding formulation of the dual value function is then

$$G(\mathbf{b}) = \max\{\mathbf{b}^T\mathbf{y} \mid \hat{A}^T\mathbf{y} \leq \hat{\mathbf{c}}, \mathbf{y} \geq 0\}.$$

The proof below is stated in [5, p. 216] and has been adjusted to our problem formulation in terms of the notation used.

Proof: Let $\hat{X}(\mathbf{b}) \neq \emptyset$ and $\mathbf{y}^* \in Y^*(\mathbf{b})$. From strong duality (4.8), it follows that $\mathbf{b}^T\mathbf{y}^* = F(\mathbf{b})$. Consider now a vector $\mathbf{b}+\mathbf{u} \in \mathbb{R}^m$ such that $\hat{X}(\mathbf{b}+\mathbf{u}) \neq \emptyset$. Due to weak duality (4.7), it holds that

$$(\mathbf{b}+\mathbf{u})^T\mathbf{y}^* \leq \hat{\mathbf{c}}^T\hat{\mathbf{x}}$$

for any feasible solution $\hat{\mathbf{x}} \in \hat{X}(\mathbf{b})$. If we take the minimum over all $\hat{\mathbf{x}} \in \hat{X}(\mathbf{b}+\mathbf{u})$, we obtain

$$(\mathbf{b}+\mathbf{u})^T\mathbf{y}^* \leq F(\mathbf{b}+\mathbf{u}).$$

Accordingly,

$$
\begin{aligned}
& (\mathbf{b}+\mathbf{u})^T\mathbf{y}^* - \mathbf{b}^T\mathbf{y}^* \leq F(\mathbf{b}+\mathbf{u}) - F(\mathbf{b}) \\
\Leftrightarrow \quad & F(\mathbf{b}+\mathbf{u}) \geq F(\mathbf{b}) + \mathbf{u}^T\mathbf{y}^*.
\end{aligned}
$$

From this, we conclude that $\mathbf{y}^* \in \partial F(\mathbf{b})$.

Proving the converse, we assume that $\mathbf{y}^* \in \partial F(\mathbf{b})$ which implies that

$$F(\mathbf{b}) + ((\mathbf{b}+\mathbf{u}) - \mathbf{b})^T\mathbf{y}^* \leq F(\mathbf{b}+\mathbf{u}) \tag{4.9}$$

for all $\mathbf{b}+\mathbf{u} \in \mathbb{R}^m$ such that $\hat{X}(\mathbf{b}+\mathbf{u}) \neq \emptyset$. Choosing any $\hat{\mathbf{x}} \in \hat{X}(\mathbf{b}+\mathbf{u})$ and letting $\hat{A}\hat{\mathbf{x}} = \mathbf{b}+\mathbf{u}$, it holds that $F(\mathbf{b}+\mathbf{u}) \leq \hat{\mathbf{c}}^T\hat{\mathbf{x}}$. With (4.9), we obtain

$$(\hat{A}\hat{\mathbf{x}})^T\mathbf{y}^* = (\mathbf{b}+\mathbf{u})^T\mathbf{y}^* \leq F(\mathbf{b}+\mathbf{u}) - F(\mathbf{b}) + \mathbf{b}^T\mathbf{y}^* \leq \hat{\mathbf{c}}^T\hat{\mathbf{x}} - F(\mathbf{b}) + \mathbf{b}^T\mathbf{y}^*$$

which is true for all $\hat{\mathbf{x}} \geq 0$. From this, we can extract that

$$(\hat{A}^T\mathbf{y}^* - \hat{\mathbf{c}})^T\hat{\mathbf{x}} \leq \mathbf{b}^T\mathbf{y}^* - F(\mathbf{b}) \;\; \forall\, \hat{\mathbf{x}} \geq 0,$$

and it follows that $\hat{A}^T\mathbf{y}^* = \hat{\mathbf{c}}^T$. Therefore, $\mathbf{y}^*$ is a dual feasible solution. Furthermore, if $\hat{\mathbf{x}} = 0$, we obtain $F(\mathbf{b}) \leq \mathbf{b}^T\mathbf{y}^*$. However, due to weak duality, every dual feasible solution $\bar{\mathbf{y}}^*$ must satisfy $\mathbf{b}^T\bar{\mathbf{y}}^* \leq F(\mathbf{b}) \leq \mathbf{b}^T\mathbf{y}^*$. This shows that $\mathbf{y}^* \in Y^*(\mathbf{b})$. □

In the literature, one can find two approaches of defining shadow prices. One is using the directional derivative of function F and has been applied in Akgül's paper about shadow prices in linear programming [1]. The other approach is defining them as the optimal solution of the dual problem [19]. In the following, we will present both approaches and proof their equivalence. Note that we will reformulate Akgül's definition according to the problem type considered in this thesis.

Definition 4.5: (*Shadow price, Akgül* [1])
Let $\mathbf{e}_i$ be the i-th unit vector. The positive shadow price of the i-th constraint is defined by

$$p_i^+ := D_{\mathbf{e}_i}F(\mathbf{b}) \; \forall i$$

and accordingly, its negative shadow price is defined by

$$p_i^- := -D_{(-\mathbf{e}_i)}F(\mathbf{b}) \; \forall i.$$

Let $\mathbf{u} \in \mathbb{R}_+^m$, then we define the positive and negative shadow price for a combination of several constraints i by

$$p_{\mathbf{u}}^+ := D_{\mathbf{u}} F(\mathbf{b})$$

and

$$p_{\mathbf{u}}^- := -D_{(-\mathbf{u})} F(\mathbf{b}).$$

In order to obtain a positive value for p_i^-, $D_{(-\mathbf{e}_i)} F(\mathbf{b})$ is multiplied by -1.

The positive shadow price of the i-th constraint represents the rate of change in F for a marginal increase in b_i and similarly, the negative shadow price is showing the rate of change in F for a marginal decrease in b_i. In the context of conservation planning, those shadow prices describe the marginal costs of increasing or decreasing the target for a certain feature. Additionally, $p_{\mathbf{u}}$ describes the change of F by increasing or decreasing a 'package' of constraints, as decision makers may be interested in how much a combination of several targets might influence total cost.

Remark 4.3: If $\mathbf{u} \in \mathbb{R}^m$, as in Akgül [1], it may happen that some elements of $\mathbf{u}$ are positive and some negative. In this case, the interpretation of positive and negative shadow prices with an increase or decrease in the right-hand side would not make sense anymore. After all, even the positive shadow price $p_{\mathbf{u}}^+$ could take negative values and analogously, $-p_{\mathbf{u}}^-$ can be positive. Example 4.2 below, shows such a case. In this regard, it is better to restrain the vector so that $\mathbf{u} \in \mathbb{R}_+^m$. Then, the shadow prices of any type will always take positive values, since F is monotonically increasing. In this thesis, we will not use Akgül's definition but another one which will be given directly.

The definition given in 4.5, refers only to LP in canonical form where prices are always considered having a non-negative value. Having a look at the objective function of the dual problem (4.4), one can interpret the dual variable $\mathbf{y}$ as an indicator of how 'important' the respective constraint i is for the optimal value [19]. Therefore, we can give an alternative definition for shadow prices using dual variables.

Definition 4.6: (*Shadow price, dual variable* [19])
Consider the dual linear optimization problem from Definition 4.2. We define the shadow price of the i-th constraint as the optimal solution of the DLP

$$p_i := y_i^* \ \forall i,$$

and similarly the shadow price of a combination of several constraints as

$$p_{\mathbf{u}} := \mathbf{u}^T \mathbf{y}^*,$$

with $\mathbf{u} \in \mathbb{R}^m$.

Here, a distinction between positive and negative shadow prices would also be possible for the perturbation of a single constraint. Depending on if there is either an increase or decrease in the value of b_i, we would call the corresponding y_i^* the positive or negative shadow price. Let us now poof the equivalence of the two shadow price definitions 4.5 and 4.6.

Corollary 4.2: Let $\mathbf{u} \in \mathbb{R}^m$. Then,

$$D_{\mathbf{u}} F(\mathbf{b}) = \mathbf{u}^T \mathbf{y}^*.$$

Proof: Consider Definitions 4.5 and 4.6. Let $\mathbf{x}^*$ be the optimal solution of the primal and $\mathbf{y}^*$ the optimal solution of the corresponding dual problem. Thus,

$$F(\mathbf{b}) = \mathbf{c}^T \mathbf{x}^*.$$

Furthermore, let $\varepsilon > 0$ be small enough so that $\mathbf{y}^*$ stays optimal. Then,

$$\begin{aligned} D_{\mathbf{u}} F(\mathbf{b}) &= \lim_{\varepsilon \to 0^+} \frac{F(\mathbf{b} + \varepsilon \mathbf{u}) - F(\mathbf{b})}{\varepsilon} \\ &\overset{(4.8)}{=} \lim_{\varepsilon \to 0^+} \frac{(\mathbf{b} + \varepsilon \mathbf{u})^T \mathbf{y}^* - \mathbf{c}^T \mathbf{x}^*}{\varepsilon} \\ &= \lim_{\varepsilon \to 0^+} \frac{\mathbf{b}^T \mathbf{y}^* + \varepsilon \mathbf{u}^T \mathbf{y}^* - \mathbf{c}^T \mathbf{x}^*}{\varepsilon} \\ &\overset{(4.8)}{=} \lim_{\varepsilon \to 0^+} \frac{\mathbf{c}^T \mathbf{x}^* + \varepsilon \mathbf{u}^T \mathbf{y}^* - \mathbf{c}^T \mathbf{x}^*}{\varepsilon} \\ &= \lim_{\varepsilon \to 0^+} \frac{\varepsilon \mathbf{u}^T \mathbf{y}^*}{\varepsilon} \\ &= \mathbf{u}^T \mathbf{y}^*. \end{aligned}$$

Particularly, with $\mathbf{u} := \mathbf{e}_i$, we obtain the shadow price of a single constraint

$$D_{\mathbf{e}_i} F(\mathbf{b}) = \mathbf{e}_i^T \mathbf{y}^* = y_i^*.$$

This proves that both approaches, defining shadow prices as the directional derivative of $F(\mathbf{b})$ and as the optimal dual solution of the same underlying optimization problem, are equivalent. $\square$

Corollary 4.3: According to Gal [10, p. 61], it holds that in case an optimal solution is not primal degenerate

$$y_i^* = p_i^+ = p_i^- \;\; \forall\, i.$$

If we already solved a LP with the simplex algorithm as presented in Chapter 3.2, we can easily extract the shadow prices from the final optimal tableau, as the following example shows.

Example 4.1: (*Shadow prices from the simplex tableau*)
Consider the final optimal simplex tableau in Table 8. The shadow prices can be found in the second row as the cost coefficients of the slack variables, i.e. $\mathbf{y}_1^* = (0, 75)^T$. Since the optimal solution of the tableau is degenerate, there are multiple shadow prices. Another one can be extracted from Table 9, i.e. $\mathbf{y}_2^* = (93.75, 75)^T$.

Example 4.2: (*'Package' shadow price*)
Consider the relaxed optimization problem from Example 2.3, with a new vector of targets $\mathbf{b} := (2, 7)^T$. Even though we changed the RHS, the optimal dual solution $\mathbf{y}^* = (0, 75)^T$ and the optimal objective function value $z^* = F(\mathbf{b}) = \$475$ stay the same. Define $\mathbf{u}^T := (1, -2)$. Then,

$$\begin{aligned} p_{\mathbf{u}}^+ &= D_{\mathbf{u}}F(\mathbf{b}) &&= D_{\left(\begin{smallmatrix} 1 \\ -2 \end{smallmatrix}\right)} F\left(\begin{pmatrix} 2 \\ 7 \end{pmatrix}\right) \\ &= \mathbf{u}^T\mathbf{y}^* &&= (1, -2) \cdot \begin{pmatrix} 0 \\ 75 \end{pmatrix} \\ &= -150 \end{aligned}$$

This is an example for a positive package shadow price taking a negative value. The interpretation of $p_{\mathbf{u}}^+ = -150$ is that if $\mathbf{b}$ gets perturbed by a marginal factor $\varepsilon > 0$, the optimal objective function value will change by the amount of $\varepsilon \cdot (-150)$. Let $\varepsilon := 0.5$. Then, the difference between the optimal objective function values is

$$\begin{aligned}\Delta\mathbf{z}^* &= F(\mathbf{b}+\varepsilon\mathbf{u}) - F(\mathbf{b})\\ &= F\left(\begin{pmatrix}2.5\\6\end{pmatrix}\right) - F\left(\begin{pmatrix}2\\7\end{pmatrix}\right)\\ &= 400 - 475\\ &= \varepsilon\mathbf{u}^T\mathbf{y}^*\\ &= 0.5\cdot(-150)\\ &= -75\end{aligned}$$

As in Definition 4.5, changes in the objective function value caused by the variation of the RHS, can only be described by $\mathbf{y}^*$, if the perturbation is small enough, i.e. marginal. Even though geometrically, a small change in $\mathbf{b}$ generates a shift of $\mathbf{x}^*$, the optimal solution remains being at the intersection of the same restricting lines. This means that the basic variables remain, but they take different values. If the perturbation is too big, the composition of the basic variables changes and with them the dual optimum. This may happen even with a unit change of the RHS. Therefore, perturbations which are not marginal cannot be described by the previously calculated shadow prices anymore. This is a very important fact, since handling shadow prices incautiously can lead to wrong results and conclusions. The example below illustrates this observation.

Example 4.3: (*Shadow price*)
Consider the relaxed version of the optimization problem (2.17) from Chapter 2, with $x_i \in [0,1]\ \forall i$. The optimal solution is

$$\mathbf{x}^* = (1, 0.25, 0)^T,$$

with total cost $z^* = \$475$. For the dual optimum, we obtain

$$\mathbf{y}^* = (0, 75)^T$$

as shadow prices for species 1 and 2 respectively. For simplicity, let us consider only positive perturbations of the RHS. The value $y_2^* = 75$ tells us that for marginal changes $\varepsilon > 0$ in b_2, total cost increase by the amount of $\varepsilon \cdot y_2^*$.

If we increase the target for species 2 by one unit, i.e. $b_2' = 8$, the new optimal solution is

$$\mathbf{x}^{*'} = (1, 0.5, 0)^T,$$

with $z^{*'} = \$550$. The basis variables did not change, only the value of x_2^* increased by 0.25, which means that the change in the optimal objective function value

$$\Delta z^* = \$550 - \$475 = \$75$$

can be described by $1 \cdot y_2^* = \$75$.

However, if we choose $b_2'' = 11$, total cost are $z^{*''} = \$900$. Thus,

$$\Delta z^* = \$900 - \$475 = \$425,$$

which is not equal to $4 \cdot \$75 = \$300$.

This result can be explained by a change in the basis. In this case, the perturbation of $\varepsilon = 4$ was not marginal anymore and generated a new combination of the basis variables. If this happens, $y_2^* = \$75$ can no longer be used to describe the change in z^*.

As the example shows, there is a certain range for each shadow price in which it is valid and the basis would not change. These ranges can be calculated using linear parametric optimization, the topic of which will be covered in the following section.

4.1.2 Linear Parametric Optimization

In the previous section, we analysed, how marginal perturbations of the RHS of an LP influence the optimal objective function value. In practice, conservation planners are interested in bigger target variations, but this could lead to a change in the basis of an optimal simplex tableau.

Graphically, this means that for marginal perturbations, the optimal point changes but stays at the vertex of the same restricting lines. For bigger perturbations, the lines of intersection can change. Therefore, the difference in total cost caused by a non-marginal perturbation cannot be described by the shadow prices of the initial solution anymore. In this case, we can use linear parametric optimization techniques in order to determine the optimality ranges of a solution $\mathbf{x}^*(t)$ for different choices of a parameter t.

The explanations in this section are all based on Chapter 2.2 of Nickel, Stein and Waldmann 2011 [19]. As previously, we assume that all primal and dual optimal solutions are non-degenerate. Let us now give some basic definitions first.

Definition 4.7: (*Perturbation of the right-hand side*)
Let $t \in T \subseteq \mathbb{R}$ and $\boldsymbol{\beta} \in \mathbb{R}^m$. Furthermore, let $\mathbf{b} \in \mathbb{R}^m$ be the RHS of a LP. Then, we define the perturbation of $\mathbf{b}$ by a parameter t as

$$\mathbf{b}(t) := \mathbf{b} + t\boldsymbol{\beta}.$$

In the following, we define w.l.o.g. that $\boldsymbol{\beta} := \mathbf{e}_i$, where $\mathbf{e}_i$ is the i-th unit vector.

Definition 4.8: (*Primal linear parametric optimization problem*)
Let $t \in T$. The parametric formulation of the primal linear optimization problem (4.1) is

$$\begin{aligned} (P_t) \quad & \min_{\mathbf{x}(t)} \mathbf{c}^T\mathbf{x}(t) \\ & \text{s.t.} \quad A\mathbf{x}(t) \geq \mathbf{b}(t) \\ & \qquad \mathbf{x}(t) \geq 0. \end{aligned} \tag{4.10}$$

According to this, we can define the set of primal feasible parametric solutions by

$$X(\mathbf{b}(t)) := \{\mathbf{x}(t) \in \mathbb{R}^n \,|\, A\mathbf{x}(t) \geq \mathbf{b}(t),\ \mathbf{x}(t) \geq 0\},$$

the optimal primal parametric objective function value by

$$F(\mathbf{b}(t)) := \{\mathbf{c}^T\mathbf{x}(t) \,|\, \mathbf{x}(t) \in X(\mathbf{b}(t))\}$$

and the set of primal optimal parametric solutions by

$$X^*(\mathbf{b}(t)) := \{\mathbf{x}(t) \in X(\mathbf{b}(t)) \,|\, \mathbf{c}^T\mathbf{x}(t) = F(\mathbf{b}(t))\}.$$

Geometrically, a change in t causes a parallel shift of the i-th restricting line (depending on the choice of $\boldsymbol{\beta}$). In Figure 7, it can be seen that for $t = 0$ the set of feasible solutions $X(\mathbf{b}(t))$ equals the set X defined in Equation (4.2). The point on the right-hand side marks the optimal solution. If we choose $t < 0$, the i-th restricting line moves to the lower left (dashed line). In this case, the basic solution changes because the new optimal point is the vertex of the i-th constraint and another line. This shift also changes the set of feasible solutions $X(\mathbf{b}(t))$. For big values of t, it may even happen that $X(\mathbf{b}(t)) = \emptyset$, as it is the case in Figure 7 for $t > 0$.

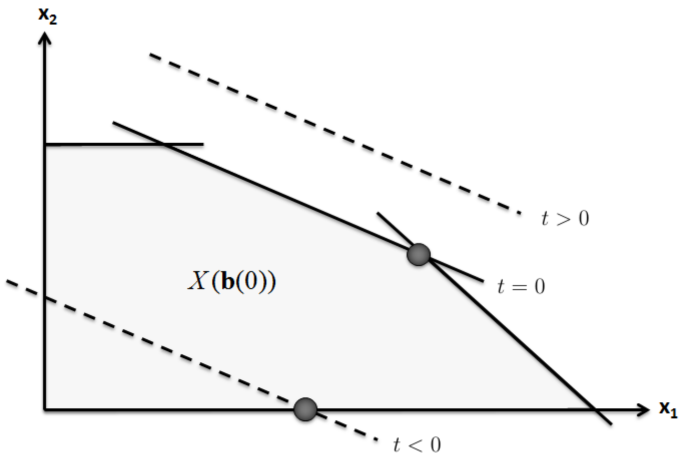


Figure 7: Effects of parameter t on the i-th restricting line [19].

The example above shows that there exist different optimality ranges for which an optimal solution has a fixed basis. The aim is now to identify these intervals of t and their corresponding sets of optimal solutions.

For this purpose, we add the vector $\boldsymbol{\beta}$ as an additional column to the initial simplex tableau (see Table 13). It is not going to influence the pivoting in any way, but at each pivot step it will be transformed by the same rules as the rest of the tableau.

	$\mathbf{x}(t)$	$\tilde{\mathbf{x}}(t)$		
z	$-\mathbf{c}$	$\tilde{\mathbf{c}}$	0	0
$\tilde{\mathbf{x}}$	$-A$	I	$-\mathbf{b}$	$-\boldsymbol{\beta}$

Table 13: Initial dual parametric simplex tableau.

After solving this tableau, we obtain a basis solution $\mathbf{x}(t) = \mathbf{x} + t\boldsymbol{\beta}'$, where $\boldsymbol{\beta}'$ is the transformed initial $\boldsymbol{\beta}$. Let B be the corresponding basis. The point $\mathbf{x}(t)$ is not necessarily primal feasible for all choices of t, since their might exist a $j \in B$ so that $x_j(t) < 0$. Therefore, we have to analyse for which values of t it holds that $x_j(t) \geq 0$ for all $j \in B$.

Provided that this is the case for $t^* \in [a, b]$, we obtain the optimal solutions $\mathbf{x}^*(t^*)$, which do all have the same basis. Note that for $t^* = a$ and $t^* = b$, the optimal solution is primal degenerate. For $t \in T \backslash [a, b]$, the corresponding tableau is primal infeasible but dual feasible. For values t from the edge of an optimality range, it is thus necessary, to carry out additional pivot steps, until the tableau is primal and dual feasible again. Thereby, we obtain new solutions $\mathbf{x}^{'}(t)$ (with a new basis), for which we can determine the respective optimality range again. This procedure is repeated, until all optimal solutions have been found for all $t \in T$.

An illustrative example of the method can be found in [19, p. 73 ff.].

4.1.3 Shadow Prices under Degeneracy

In Section 3.2.2, we already explained primal degeneracy. Here, we are going to focus on its effects on shadow prices. In doing so, we will define the 'true' shadow price and give some recommendations for further references that cover this subject.

As we have seen previously, degeneracy causes multiple dual optimal solutions, i.e. there exist more than one shadow price for a primal optimal solution. An example of the consequences can be seen in Figure 8. The considered maximization problem is restrained by three less than or equal constraints c_1, c_2 and c_3 (black lines). The optimum is denoted by P_1. It can be seen that the solution is degenerate, since one of the constraints c_2 or c_3 is redundant.

Let us assume that we obtained the shadow price by defining P_1 as the intersection of constraints c_1 and c_2. A decrease in the RHS of c_2, leads to a left shift of the respective line (green line). The new optimum P_2 can also be defined by c_1 and c_2 which means it has the same basis, and the change in the objective function value can be described by the obtained shadow price.

However, if we increase the RHS of c_2, this causes a shift to the right (blue line). The new optimum P_3 can only be defined by constraints c_1 and c_3. In this case, we cannot use the assumed shadow price, but the shadow price we would have obtained by defining P_1 through constraints c_1 and c_3.

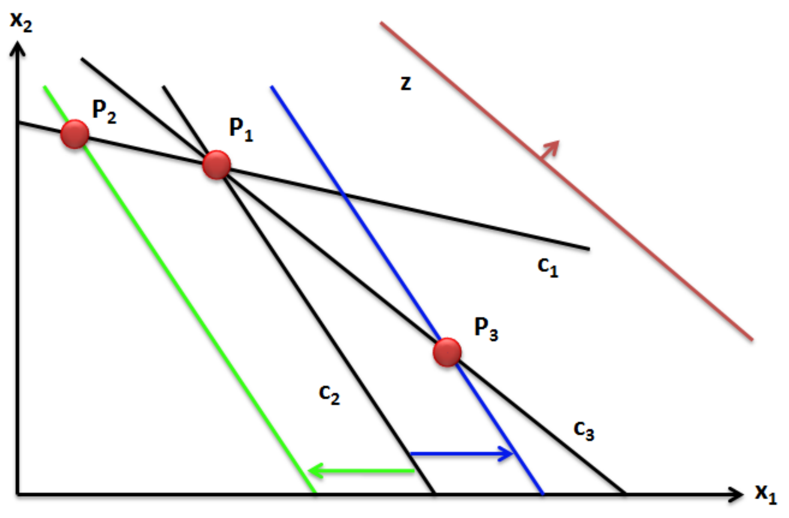


Figure 8: Example of the effect of degeneracy on shadow prices.

This small example shows that under degeneracy, the shadow price of a constraint is not unique but rather 'two-sided' (one for an increase and one for a decrease in the RHS). Looking back at Akgül's shadow price definition in Definition 4.5, we already characterized it to be two-sided, by defining a positive and a negative shadow price. In Corollary 4.3, we furthermore stated, that in the non-degenerate case, the shadow price is unique, i.e. both sides are equal.

Thus, under degeneracy the question arises, which shadow price should be used for sensitivity analysis, i.e. which is the 'true' shadow price?

Definition 4.9: (*True shadow price*, [14])
Consider a minimization problem with less than or equal constraints and let k be the number of shadow prices y_i^* of the i-th constraint of a primal degenerate optimal solution. Then, we define the true shadow price by

$$p_i^+ := \max_k \{y_{i,k}^*\},$$

for an increase in the RHS and

$$p_i^- := \min_k \{y_{i,k}^*\},$$

for a decrease in the RHS.

Note that this definition has been adjusted to the problem type considered in this thesis. The definition given, for example, by Ho [14], Akgül [1] and Aucamp and Steinberg [2], bases on a maximization problem and is given by

$$\tilde{p}_i^+ := \min_k \{\tilde{y}_{i,k}^*\}$$

and

$$\tilde{p}_i^- := \max_k \{\tilde{y}_{i,k}^*\},$$

with $\tilde{y}_{i,k}^*$ being the shadow prices of the i-th constraint, obtained by solving a maximization problem.

Proposition 4.1: For a minimization problem with less than or equal constraints, both definitions are equivalent.

Proof: Consider a minimization problem with less than or equal constraints. From $\min \mathbf{c}^T\mathbf{x} = -\max -\mathbf{c}^T\mathbf{x}$, it follows that

$$-y_{i,k}^* \Leftrightarrow \tilde{y}_{i,k}^* \ \ \forall i, k$$

and furthermore, $\tilde{y}_{i,k}^* \leq 0$ for all i, k. Thus, it also holds that $\tilde{p}_i^+, \tilde{p}_i^- \leq 0$, for all i. Therefore,

$$\tilde{p}_i^+ = \min_k \{\tilde{y}_{i,k}^*\} \Leftrightarrow -\tilde{p}_i^+ = -\min_k \{-y_{i,k}^*\} = \max_k \{y_{i,k}^*\} = p_i^+$$

and analogously,

$$\tilde{p}_i^- = \max_k \{\tilde{y}_{i,k}^*\} \Leftrightarrow -\tilde{p}_i^- = -\max_k \{-y_{i,k}^*\} = \min_k \{y_{i,k}^*\} = p_i^+,$$

which proves the assertion. □

Let us now give a small example of true shadow prices.

Example 4.4: (*True shadow price*)
For the optimal solution $\mathbf{x}^* = (1, 0.25, 0)^T$ of the relaxed problem (2.17), we obtain the following true shadow prices:

$$p_1^+ = 93.75, \qquad p_1^- = 0,$$

$$p_2^+ = 75 \qquad \text{and} \quad p_2^- = 75.$$

Table 14, validates these results assuming a one by one perturbation of $\varepsilon := \pm 0.5$ of each constraint. Remember, the initial RHS is $\mathbf{b} = (3,7)^T$ and that we intent to show that

$$\Delta z^* = F(\mathbf{b} + \varepsilon \mathbf{e}_i) - F(\mathbf{b}) = \varepsilon \cdot p_i^{\pm}.$$

$\mathbf{b}'$	Δz^*		$\varepsilon \cdot p_i^{\pm}$	
$(3.5,7)^T$	$521.875 - 475$	$= 46.875$	$0.5 \cdot 93.75$	$= 46.875$
$(2.5,7)^T$	$475 - 475$	$= 0$	$-0.5 \cdot 0$	$= 0$
$(3,7.5)^T$	$512.5 - 475$	$= 37.5$	$0.5 \cdot 75$	$= 37.5$
$(3,6.5)^T$	$437.5 - 475$	$= -37.5$	$-0.5 \cdot 75$	$= -37.5$

Table 14: Validation of the true shadow price example.

In order to calculate all shadow prices from an optimal degenerate simplex tableau, Aucamp and Steinberg [2] developed the concept of the 'valid degenerate pivot row' (VDPR). A revised definition and an example can be found in Akgül [1, p. 429 ff.] and in Gal [10, p. 63].

Knolmayer developed an algorithm which analyses the algebraic "signs of the coefficients in the degenerate rows of the optimal simplex tableaus. [...][It] stops generating additional tableaus as soon as all the information required is obtained; typically only a fraction of all optimal tableaus has to be considered" [16, p. 14]. More detailed information about this method along with some examples can be found in the respective journal article [16] and in [10].

Having now covered the topic of sensitivity analysis of LPs, we will continue with the sensitivity analysis of ILPs in the following section.

4.2 Integer Linear Optimization Problems

Sensitivity Analysis for ILPs is a lot more challenging than for LPs, taking into account the fact that even solving the optimization problem with only one fixed value of the RHS is NP-hard. In this section, we will give a brief overview over methods available along with some references for further information.

In terms of the complexity of implementation, the simplest approach of sensitivity analysis is resolving the ILP with different values for the RHS. We will use this approach later on in Chapter 5. However, this method can be very time consuming for large problem sizes.

Another approach is using the so-called value function, as we defined it in (4.3), adjusted to the ILP

$$F_{ILP}(\mathbf{b}) := \min\{\mathbf{c}^T\mathbf{x} \,|\, A\mathbf{x} + I\tilde{\mathbf{x}} = \mathbf{b},\ \mathbf{x} \in \{0,1\},\ \tilde{\mathbf{x}} \geq 0\},$$

with $I \in \mathbb{R}^{m\times m}$ being the unit matrix and $\tilde{\mathbf{x}} \in \mathbb{R}^m_+$ the slack variables. Note that this is a mixed integer linear program (MILP).

According to Guzelsoy and Ralphs [13, p. 2], a dual function $G_{ILP}(\mathbf{b})$, is a function that bounds the value function over the set $\mathbb{R}^m$, that is,

$$G_{ILP}(\mathbf{b}) \leq F_{ILP}(\mathbf{b})\ \forall\ \mathbf{b} \in \mathbb{R}^m.$$

A feasible dual function can be easily obtained for any MILP. Consider, for example, the value function of the LP relaxation of the primal problem:

$$F_{LP}(\mathbf{b}) \leq F_{ILP}(\mathbf{b})\ \forall\ \mathbf{b} \in \mathbb{R}^m,$$

taking into account a minimization problem.

The aim is to construct a dual function G_{ILP}, which closely approximates the primal value function F_{ILP}, i.e. at a point $\bar{\mathbf{b}} \in \mathbb{R}^m$ its value should be as close as possible to $F_{ILP}(\bar{\mathbf{b}})$. We can then use the dual value function in order to obtain lower and upper bounds to the primal objective function value for a modification of the RHS. In case the upper bound equals the lower bound for a fixed $\bar{\mathbf{b}}$, we can deduce that $F_{ILP}(\bar{\mathbf{b}}) = G_{ILP}(\bar{\mathbf{b}})$ [12, p. 132].

There are different methods of constructing a dual value function, using for example a cutting plane method, lagrangian relaxation or branch and bound. Guzelsoy and Ralphs give an overview of various approaches in their paper about integer programming duality [13]. More detailed descriptions can be found in Guzelsoys' dissertation [12].

However, both authors conclude that it "has so far been difficult to replicate for integer programming the functional and efficient implementations of dual methods we have become so accustomed to in the linear programming case" [13, p. 11]. Nevertheless, they also point out that with advances in computing, better implementations might be possible.

In 1988, Kim and Cho proposed “a new concept of shadow price in integer programming with rich economic interpretation which is useful in management decision” [15]. They called it the ’average shadow price’, which is based on average profitability instead of duality theory or marginal analysis.

The main aspects of their paper [15] will be summarized in the following. The authors consider a perturbation function of a maximization ILP

$$z_k(w) := \max\{\mathbf{c}^T\mathbf{x} \,|\, \mathbf{a}_j\mathbf{x} \leq \mathbf{b}_j (j \neq k), \mathbf{a}_k\mathbf{x} \leq b_k + w, \mathbf{x} \in \mathbb{Z}^n_+\},$$

where $\mathbf{a}_j$ denotes the j-th row of matrix A. This function describes the objective function value, if the k-th restriction is perturbed by an amount w.

Definition 4.10: (*Average shadow price*, [15])
The average shadow price of the k-th restriction is

$$y_k := \inf\{p \geq 0 \,|\, z_k(w) - z_k(0) - pw \leq 0 \;\forall w \geq 0\}.$$

According to [15, p. 330], the above definition is equivalent to

$$y_k = \sup_{w>0}\{(z_k(w) - z_k(0))/w\}.$$

Here, $(z_k(w) - z_k(0))/w$ is the average additional profit for a perturbation w of the k-th constraint. Another characteristic of y_k is that “the average shadow price reduces to the marginal shadow price if a convex programming problem is considered” [15, p. 331].

In order to obtain y_k, it is necessary to solve a sequence of ILPs. Depending on the problem size, this can be very time consuming. Thus, the authors provide a procedure of finding upper and lower bounds to y_k, by solving only one more ILP. For this purpose, they define the so-called critical values for function z.

Definition 4.11: (*Critical value*, [15])
A real number w is called a critical value for the function z, if $z(w') < z(w)$ for all $w' < w$. Let C be the set of all positive critical values.

Lemma 4.3: If C is empty, $y_k = 0$. Otherwise,

$$y_k = \max_{w \in C}\{(z(w) - z(0))/w\}. \tag{4.11}$$

The proof of this lemma can be found in [15, p. 332].

The largest value that solves (4.11), is called the steepest critical value and is denoted by w^*. The following theorem defines an upper and lower bound of the average shadow price.

Theorem 4.3: (*Upper and lower bound of* y_k, [15])
Let $C \neq \emptyset$ and let $\bar{z}$ be the perturbation function of the LP relaxation of the initial programming problem. For any w with $0 \leq w \leq w^*$, define the lower bound of y_k by

$$y^l := (z(w) - z(0))/w$$

and the upper bound by

$$y_1^u := (\bar{z}(w) - z(0))/w.$$

Then, y_k is in $[y^l, y_1^u]$.

The proof of this theorem is given in [15, p. 332].

According to the authors, "the Lagrangian relaxation provides a better upper bound for y_k than the LP relaxation" [15, p. 333]. Analogously to the theorem above, we can therefore define another upper bound by

$$y_2^u := (z^L(w) - z(0))/w,$$

with z^L being the perturbation function of the Lagrangian relaxation of the initial optimization problem. With this, we get $y_k \in [y^l, y_2^u] \subset [y^l, y_1^u]$.

Assuming that $\mathbf{c}$ is an integer vector, a positive value $w \leq w^*$ can be found by solving the problem

$$CP := \min\{\mathbf{a}_k\mathbf{x} - b_k \,|\, \mathbf{c}^T\mathbf{x} \geq z(0) + 1, \mathbf{a}_j\mathbf{x} \leq \mathbf{b}_j \; (j \neq k), \mathbf{x} \in \mathbb{Z}_+^n\}.$$

If the initial problem is a 0-1 optimization problem, the problem CP can be solved easily using information saved during the resolution of the initial problem [15, p. 333].

In the final part of their paper, Kim and Cho present a method that improves the upper bound by solving the LP or Lagrangian relaxation of the following profit maximization problem

$$M(p) := \max\{\mathbf{c}^T\mathbf{x} - p(\mathbf{a}_k\mathbf{x} - b_k) \,|\, \mathbf{a}_k\mathbf{x} \geq b_k + w^0, \mathbf{a}_j\mathbf{x} \leq \mathbf{b}_j \; (j \neq k), \mathbf{x} \in \mathbb{Z}_+^n\},$$

with $w^0 \leq w^*$ being a positive critical value. Furthermore, they provide an algorithm which updates the lower bound, until it converges to y_k by solving a sequence of ILPs.

Note that we can easily transform the *minimum set coverage problem* into a maximization problem, analogously to the way it has been done with the *land allocation problem* in Equation (3.1). Thus, we can also use the results given above.

However, a major drawback of this method concerning the sensitivity analysis of conservation prioritization problems is, that at least two ILPs have to be solved directly. As mentioned before, this must not always be possible for larger problems which is the main reason why there is no ILP feature ranking for two of the problems considered in the next chapter. Additionally, the smallest problem which we are going to examine, can be solved in a very short time using direct resolution methods. Thus, it is not necessary to implement the method described above, though it might be good for problems that are still small enough to solve them directly but too large already to conduct a sensitivity analysis by solving a sequence of ILPs.

Let us now apply the obtained results of the previous chapters in order to answer the main research questions of this thesis.

5 Application - Creating a Feature Ranking

In the introduction of this thesis we, formulated two central questions:

- Which features have the highest influence on the total cost of a reserve system?
- How much must/can targets be changed in order to have a high impact on already calculated total cost or none at all?

In this chapter, we seek to answer these questions for the *minimum set coverage problem*. From Chapter 4 about sensitivity analysis, we know that for linear programming problems such questions can be answered with the help of shadow prices. Since shadow prices are not defined for ILPs, such as the *minimum set coverage problem*, the question is now if we can use the shadow prices of the *land allocation problem*, its relaxation, instead?

Mathematically, this is in general not possible because we assume that an optimal solution of a continuous optimization problem is very close to an optimal solution of its corresponding integer problem. In Chapter 6, we will discuss this aspect in more detail. However, in the context of the *minimum set coverage problem*, this assumption seems to work well as we will see later. Therefore, we are going to use the shadow prices of the *land allocation problem* as an approximate approach in order to answer the questions above for the *minimum set coverage problem*.

In the following, we will introduce a methodology which creates a feature ranking according to their influences on the total cost of the reserve system. Furthermore, we will discuss some problems which can occur during this process and give possible solutions. In the last part of this chapter, we will apply the developed method to three real-world *minimum set coverage* problems of different sizes and evaluate the obtained results.

5.1 Methodology

The shadow prices of the *land allocation problem* provide an indication of how much the correspondent features influence the objective function value.

Thus, the idea is to rank the features according to their shadow prices from highly sensitive (high influence) to little sensitive (low influence).

In order to test the credibility of the result, we create another ranking with Marxan results and - if possible - with the ILP solutions. Since we cannot work with shadow prices for the latter two rankings, we proceed as follows:

Algorithm 5.1: (*Pseudocode of the ILP and Marxan ranking*)

```
0. SET $d := const.$
   SOLVE initial ILP, save baseline cost $c_0$.
1. FOR $j = 1, 2, ..., m$ iterations DO
   1.1 Decrease $j$-th target by amount $d$.
   1.2 SOLVE ILP, save cost $c_j$.
   1.3 Calculate $\Delta c := c_0 - c_j$.
2. Create a ranking of all features according to their respective $\Delta c$.
3. REPEAT for new $d$.
```

The R programming code of this algorithm can be found on the CD enclosed with this thesis.

The value of constant d cannot be marginal, as it would be needed for actual shadow prices because then, we would not see any effect in the objective function value. As a matter of fact, the perturbation for the Marxan ranking must be quite big. The reason for this is the so-called Marxan error.

In Figure 9, one can see that the objective function values of solutions provided by Marxan lie within a range around the actual optimum c_0^{ILP}. If we chose a small perturbation, whose corresponding optimal objective function value c_j^{ILP} is too close to c_0^{ILP}, Marxan might provide a solution with total cost $c_j^M > c_0^M$, even though $c_j^{ILP} < c_0^{ILP}$. Thus, the constant d must be chosen big enough to avoid this case (see Figure 10).

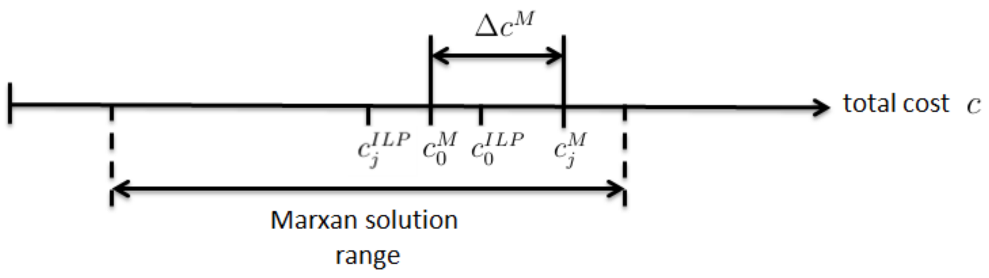


Figure 9: If the perturbation is too small, c_0^M lies within the Marxan solution range of c_j^M. Thus, possible total cost c_j^M provided by Marxan could be greater than c_0^{ILP} although c_j^{ILP} is smaller than c_0^{ILP}.

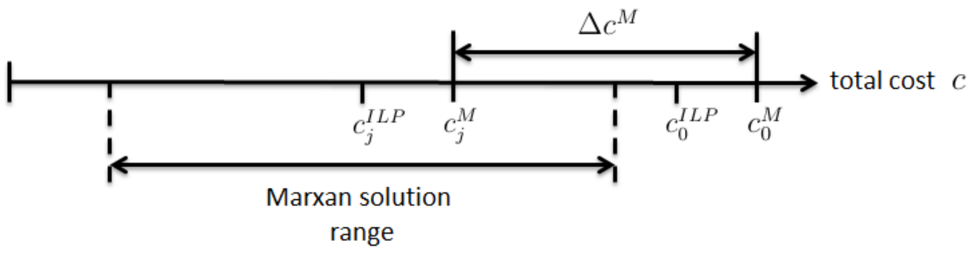


Figure 10: If the perturbation is big enough, c_0^M lies outside the Marxan solution range of c_j^M.

Furthermore, the targets must always be changed by absolute numbers, i.e. d cannot be a percentage value. Otherwise, the results may not be as expected, as the following example shows.

Example 5.1: (*Percentage vs. absolute number perturbation*)
For one of the data sets which we will introduce later, we noticed for two features, say species 1 and species 2, that for a percentage perturbation of their targets, the difference in total cost was higher for the least sensitive species 2 than for the most sensitive species 1.

The reason for this was, that decreasing the target for species 1 by 0.05%, meant reducing the total amount of individuals by approx. 50, whereas for species 2, 0.05% already corresponded to about 1657 individuals. So even though species 2 was less sensitive than species 1 (i.e. the effect on total cost should have been lower than for species 1), the high amount of reduced individuals compensated this fact and made the change in total cost higher.

This effect did not occur when the targets were decreased by an absolute number of individuals.

It must also be taken into account that for the ILP and the Marxan ranking, the range of d depends on the lowest target value. In our analysis, we always set the target to a certain percentage of the total representation of a feature. Thus, we sought to protect e.g. 30% of all individuals of each species. If there is for example a very common species, the target is respectively high, and can accordingly be decreased by a high amount of individuals. But this amount must not exceed the target value of the rarest species in order to avoid negative targets.

As discussed in Chapter 4, shadow prices are only valid for marginal changes in the RHS. This means that in terms of the necessarily high perturbations for the Marxan ranking, we have to show that the current shadow price ranking is stable towards big changes in the RHS. For this purpose, we conduct a stability check for the dual variables. The programming code of the following algorithm, written in R, is saved on the CD that is enclosed with this thesis.

Algorithm 5.2: (*Pseudocode of the duals stability check*)

```
0. SET $d := const.$
   SOLVE initial LP, create feature ranking.
1. FOR $j = 1, 2, ..., m$ iterations DO
   1.1 Decrease $j$-th target by amount $d$.
   1.2 SOLVE LP, save shadow price $s_j$.
2. Create a ranking of all features according to their respective $s_j$.
3. REPEAT for new $d$.
```

If there are no major changes in the shadow price feature rankings, we can assume the LP, ILP and Marxan rankings to be comparable. In more detail, this means that the shadow prices react more or less evenly towards changes in the targets, i.e. there is no feature that reacts for example alternately very sensitive, not sensitive at all and then sensitive again for different target ranges.

Regarding the Marxan ranking, it must also be taken into account that for rare species, the perturbations might have to be so big that the new targets are close to zero. In this case, it can happen that the simplex algorithm returns zero shadow prices. For the duals stability check this means that a reasonable ranking of the duals would not be possible anymore. Due to the Marxan error, the big perturbations can thus be seen like a 'shift' of the target changes for the duals stability check in order to make the algorithm react as sensible as the simplex.

5.2 Application

In this section, we will apply the above presented methodology to three real-world conservation prioritization problems. Table 15 gives an overview of the main information about the Marine Protected Area (MPA), the Tridacna Crocea (TCR) and the Tasmania (TAS) data set. The complete underlying data of this chapter can be found on the CD enclosed with this thesis.

Data set	**MPA**	**TCR**	**TAS**
Planning units	100	1435	1723
Features	10	22	17
Solved with LPSolve ILP	✓		
Solved with LPSolve LP	✓	✓	✓
Solved with Marxan	✓	✓	✓

Table 15: Data set overview.

For the MPA data set, it is possible to solve the corresponding *minimum set coverage problem* directly. However, the larger data sets TCR and TAS are already too big to solve them with exact resolution methods in reasonable computing time.

For all data sets, we ran the *land allocation problem* and for the MPA data set also the *minimum set coverage problem* on the open source solver LPSolve [4]. Marxan has been used as a heuristic approach for the resolution of the *minimum set coverage problems*. Here, we set the calibration parameters to

0 for the boundary length (because we did not take it into account) and to 10 for the species penalty factor (see [11, p. 98 ff.]), so that all the targets would be met in the reserve generated.

To be able to work with great perturbations for the Marxan ranking, it was necessary to choose relatively high targets. The target values for all three data sets can be found in Table 16.

Feature ID	**MPA 60%**	**TCR 67%**	**TAS 57%**
1	802.2	12184.10921	
2	753.6	11708.66411	
3	729	11155.89499	
4	834.6	11167.38754	
5	873	10346.41173	
6	624	12192.70254	
7	952.2	12962.71452	
8	774	13014.19448	
9	865.2	12789.3767	
10	1122	10540.95469	629906.7359
11		11693.93743	11256.58015
12		13142.20386	16923.62827
13		12918.15188	21782.762
14		14163.30853	18770.55671
15		13323.59393	44674.92819
16		10844.97206	34914.82193
17		12235.03702	50012.32163
18		10053.42712	71678.4257
19		12448.86853	59811.02374
20		11016.33255	39963.21131
21		11491.97427	19070.09415
22		11675.85018	25930.24384
23			27739.2742
24			25427.29452
25			27009.54411
26			10170.99598

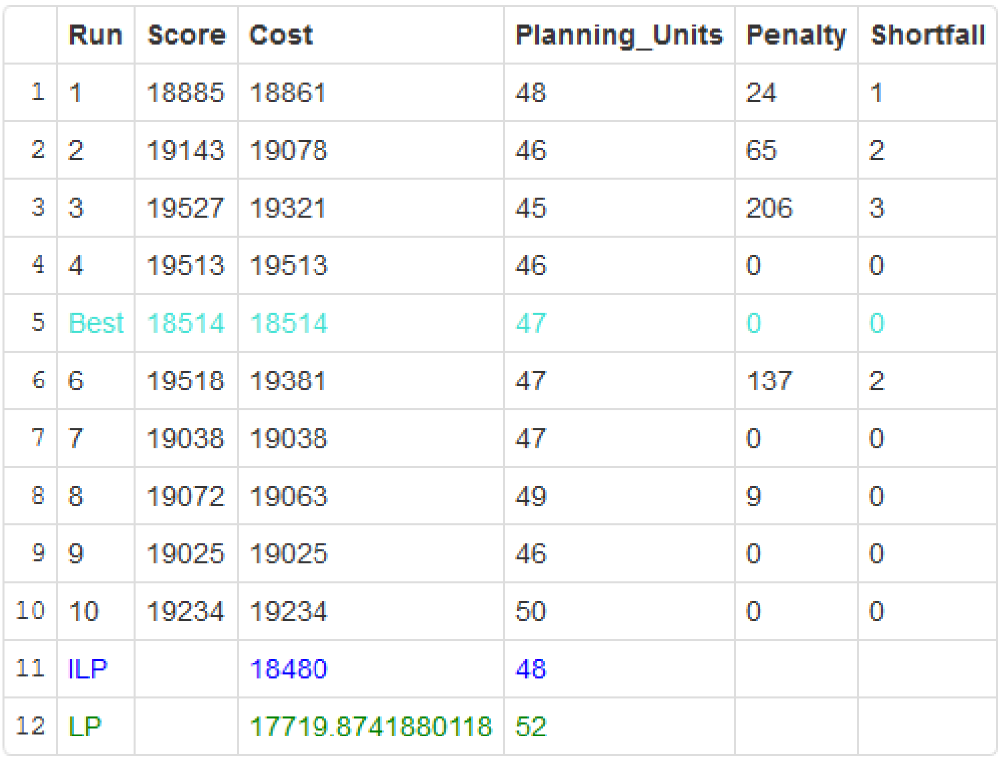
Table 16: Targets of the MPA, TCR and TAS data set. The percentage values state the share of the total amount of individuals.

Let us now have a closer look at the results of the individual data sets.

Marine Protected Area (MPA) Data Set

The MPA data set is the smallest of the three considered data sets in this thesis. It consists of 100 planning units (sites) and considers 10 species (features). As shown in Table 15, we solved the corresponding ILP and LP using LPSolve and Marxan.

The results of the different algorithms are given in the solution table of the Marxan online app [28] in Figure 11. LPSolve calculated total cost of $18480, selecting 48 planning units for the ILP and $17719.87, selecting 52 planning units for the LP, whereas the Marxan best solution was $18514, with 47 planning units selected for conservation. The LP solution is the cheapest because the algorithm allows the variables to take non-integer values which means that it is also possible to conserve only a fraction of a site.



	Run	Score	Cost	Planning_Units	Penalty	Shortfall
1	1	18885	18861	48	24	1
2	2	19143	19078	46	65	2
3	3	19527	19321	45	206	3
4	4	19513	19513	46	0	0
5	Best	18514	18514	47	0	0
6	6	19518	19381	47	137	2
7	7	19038	19038	47	0	0
8	8	19072	19063	49	9	0
9	9	19025	19025	46	0	0
10	10	19234	19234	50	0	0
11	ILP		18480	48		
12	LP		17719.8741880118	52		

Figure 11: MPA solution table of the Marxan online app [28].

Figures 12 - 14 show the conservation maps with the selected planning units marked in green or blue. It can be seen that all three approaches chose similar planning units for preservation. The different colour shades in Figure 13, indicate the amount of the corresponding planning unit selected for protection (dark blue indicates that it has been selected entirely).

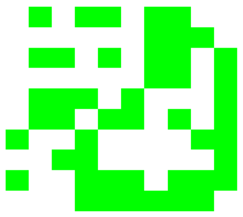


Figure 12: MPA ILP map [28].

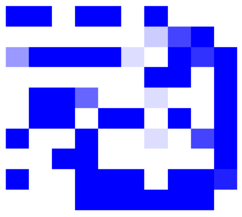


Figure 13: MPA LP map [28].

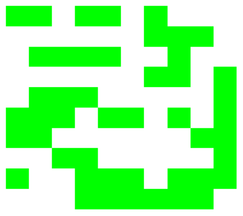


Figure 14: MPA Marxan best solution map [28].

The shadow price ranking for this data set is shown in column “LP 0” of Figure 17 (see page 73). We can see that according to the ranking, species 5, 6 and 4 are the most sensible. If we decrease their targets, they will have the highest influence on the total cost of the reserve system.

The columns “LP -150”, “LP -200” and “LP -250” show the results of the duals stability check. The numbers give the amount of the target perturbation, e.g. in column “LP -150”, all targets have been decreased one by one by 150 units. It can be seen that there are only slight changes in the ranking and that also species 9 happens to be quite sensitive. The ILP and Marxan rankings (columns “ILP -200, -250, -300” and “Marxan -200, -400, -600”) give similar results. This means that for the MPA data set shadow prices can be used for the creation of a feature ranking.

Tridacna Crocea (TCR) Data Set

The TCR data set considers 22 features in 1435 sites. Total cost of the optimal LP solution is \$1,534,888, whereas Marxan with 100 runs and 10 million iterations calculated total cost of \$1,578,492.38. According to the shadow price ranking (“LP 0”) in Figure 18 on page 74, features 22, 20, 17 and 1 are the most sensitive and species 19, 9 and 11 the least sensitive. However, the duals stability check (“LP -300, -600, -900”) shows that the shadow price for feature 17 (marked blue) is not stable. This explains its different position in the Marxan ranking.

Furthermore, the “NA”-value next to feature 11 in the last column means that due to the Marxan error, the difference in total cost was negative. This makes it impossible to rank this particular feature, though it is likely to be low sensitive with the two Marxan solutions appearing to be so ’close’ to each other.

In summary, one can say that for features with stable duals the LP solver and Marxan provide similar rankings.

Tasmania (TAS) Data Set

By taking into account 1723 planning units and 17 features, the Tasmania (TAS) data set is the largest considered in this thesis. Solving it with a 57% target, gives total cost of \$8,895,725,027 for the LP and \$9,307,419,702 for the ILP solved with Marxan. The respective map of the selected planning units can be seen in Figures 15 and 16.

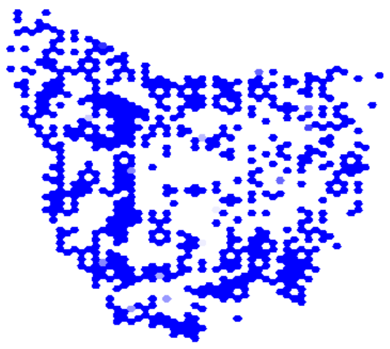


Figure 15: TAS LP map [28].

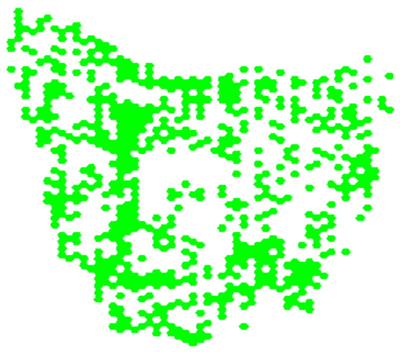


Figure 16: TAS Marxan best solution map [28].

The rankings in Figure 19 at the end of this chapter show that the dual variables of this data set are extremely stable (cf. columns “LP 0, -300, -600, -900”). Due to this, features 21, 26 and 23 are the most sensitive in all rankings and even their positions do not change much. Features 22, 17 and 10 appear to be the least sensitive, which holds for the shadow price ranking as well as the Marxan rankings.

The zeros next to features 20 and 22 in column “LP -5000” signify that in their cases, the shadow price is 0 which means that both of them hold the same position within the ranking.

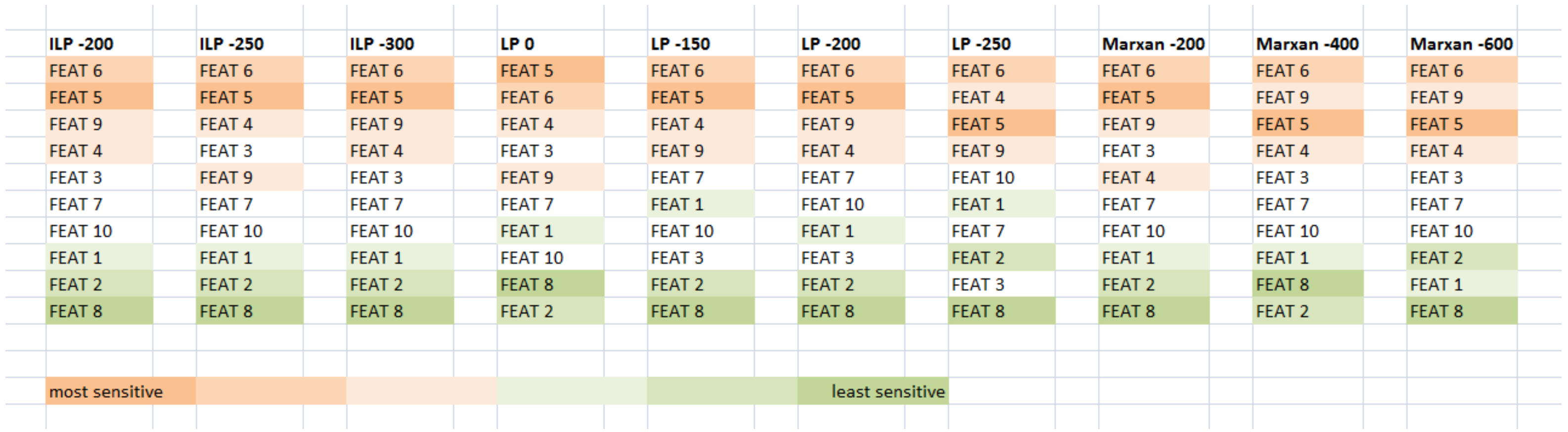
With the above considerations, we answered the first question stated at the beginning of this chapter: by using shadow prices we can create a feature ranking according to their influence on total cost. The second question, however, can only be answered for the non-integer problem, the *land allocation problem*.

Here, the optimality ranges of solutions $\mathbf{x}^*(t)$, as we introduced them in Chapter 4.1.2, also indicate the optimality ranges of the corresponding dual variables. For many problem solvers, such as LPSolve, there exists a command which displays the shadow prices along with their ranges.

From Chapter 4.1.1, we know that we can calculate the change in total cost by multiplying the change in a feature’s target by its corresponding shadow price. Thus, if we now wanted to know how much we could change a target without influencing total cost, the corresponding feature’s shadow price would have to be zero and the perturbation would have to be within the optimality range of the dual solution.

For the ILP (the *minimum set coverage problem*), shadow prices can indeed be used as an estimation of the feature’s sensitivity, but their actual values are of no use when it comes to calculating the difference in total cost for changes in the targets. The reason for this is, that the results would be too inaccurate, since the shadow price values are only correct for the relaxed problem.

A more detailed discussion of the results of this section will be part of the subject of the following chapter.

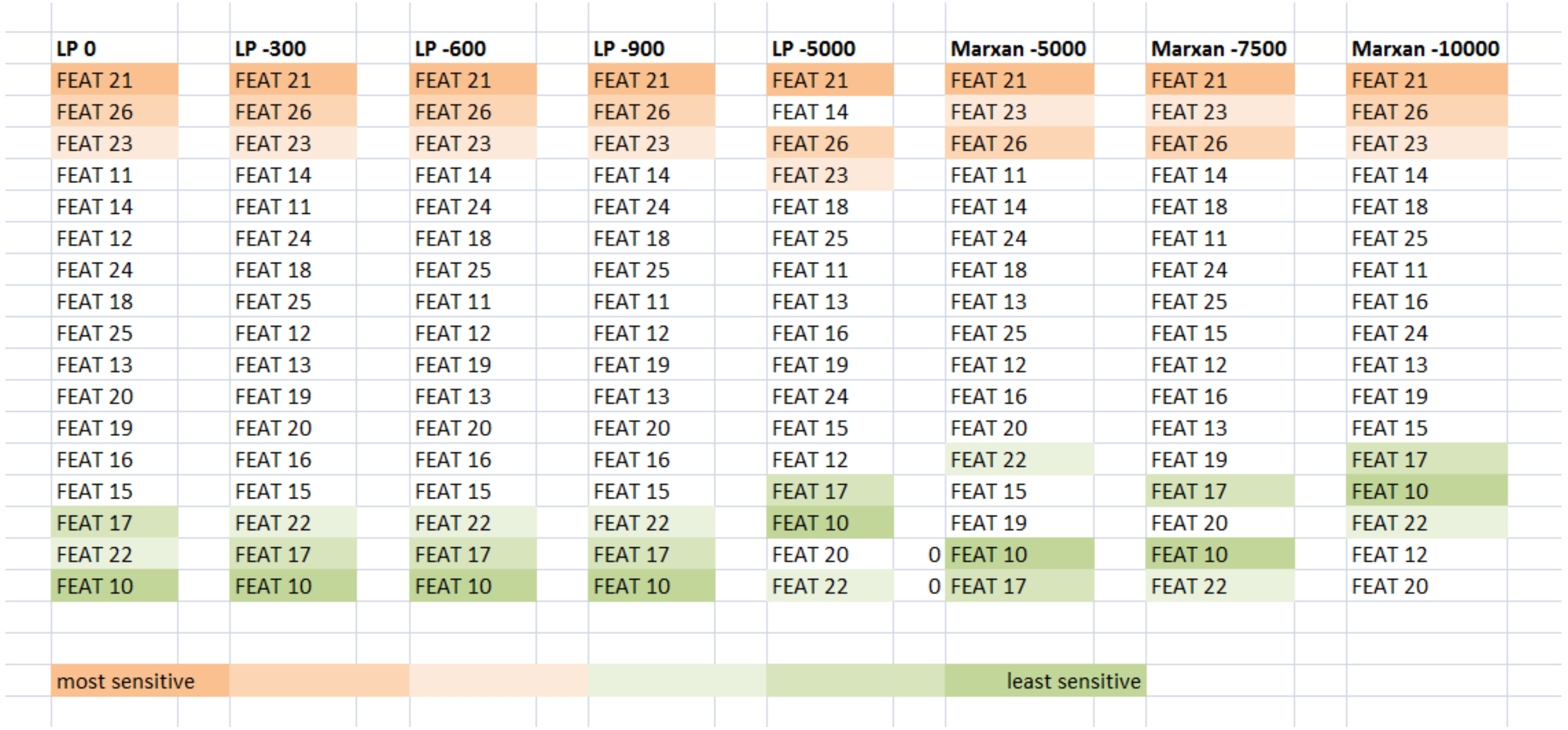


ILP -200	ILP -250	ILP -300	LP 0	LP -150	LP -200	LP -250	Marxan -200	Marxan -400	Marxan -600
FEAT 6	FEAT 6	FEAT 6	FEAT 5	FEAT 6	FEAT 6	FEAT 6	FEAT 6	FEAT 6	FEAT 6
FEAT 5	FEAT 5	FEAT 5	FEAT 6	FEAT 5	FEAT 5	FEAT 4	FEAT 5	FEAT 9	FEAT 9
FEAT 9	FEAT 4	FEAT 9	FEAT 4	FEAT 4	FEAT 9	FEAT 5	FEAT 9	FEAT 5	FEAT 5
FEAT 4	FEAT 3	FEAT 4	FEAT 3	FEAT 9	FEAT 4	FEAT 9	FEAT 3	FEAT 4	FEAT 4
FEAT 3	FEAT 9	FEAT 3	FEAT 9	FEAT 7	FEAT 7	FEAT 10	FEAT 4	FEAT 3	FEAT 3
FEAT 7	FEAT 7	FEAT 7	FEAT 7	FEAT 1	FEAT 10	FEAT 1	FEAT 7	FEAT 7	FEAT 7
FEAT 10	FEAT 10	FEAT 10	FEAT 1	FEAT 10	FEAT 1	FEAT 7	FEAT 10	FEAT 10	FEAT 10
FEAT 1	FEAT 1	FEAT 1	FEAT 10	FEAT 3	FEAT 3	FEAT 2	FEAT 1	FEAT 1	FEAT 2
FEAT 2	FEAT 2	FEAT 2	FEAT 8	FEAT 2	FEAT 2	FEAT 3	FEAT 2	FEAT 8	FEAT 1
FEAT 8	FEAT 8	FEAT 8	FEAT 2	FEAT 8	FEAT 8	FEAT 8	FEAT 8	FEAT 2	FEAT 8

Figure 17: Feature ranking of the Marine Protected Area (MPA) data set. The headers of the columns give the solver type (ILP, LP, Marxan) and the respective perturbation, i.e. column "LP 0" states the shadow prices of the initial problem.

LP 0	LP -300	LP -600	LP -900	Marxan -5000	Marxan -7500	Marxan -10000	
FEAT 22	FEAT 22	FEAT 20	FEAT 22	FEAT 22	FEAT 22	FEAT 22	
FEAT 20	FEAT 20	FEAT 22	FEAT 20	FEAT 20	FEAT 20	FEAT 20	
FEAT 17	FEAT 1	FEAT 1	FEAT 1	FEAT 21	FEAT 12	FEAT 21	
FEAT 1	FEAT 13	FEAT 13	FEAT 21	FEAT 13	FEAT 13	FEAT 13	
FEAT 16	FEAT 3	FEAT 21	FEAT 13	FEAT 12	FEAT 1	FEAT 1	
FEAT 13	FEAT 21	FEAT 8	FEAT 5	FEAT 1	FEAT 16	FEAT 12	
FEAT 3	FEAT 17	FEAT 5	FEAT 6	FEAT 16	FEAT 21	FEAT 5	
FEAT 7	FEAT 8	FEAT 18	FEAT 8	FEAT 6	FEAT 7	FEAT 6	
FEAT 15	FEAT 5	FEAT 6	FEAT 12	FEAT 5	FEAT 15	FEAT 16	
FEAT 21	FEAT 15	FEAT 12	FEAT 16	FEAT 15	FEAT 8	FEAT 8	
FEAT 5	FEAT 7	FEAT 16	FEAT 7	FEAT 18	FEAT 18	FEAT 7	
FEAT 8	FEAT 16	FEAT 7	FEAT 15	FEAT 7	FEAT 5	FEAT 18	
FEAT 6	FEAT 6	FEAT 14	FEAT 18	FEAT 17	FEAT 4	FEAT 10	
FEAT 12	FEAT 18	FEAT 15	FEAT 14	FEAT 10	FEAT 19	FEAT 4	
FEAT 18	FEAT 12	FEAT 3	FEAT 4	FEAT 8	FEAT 6	FEAT 17	
FEAT 2	FEAT 14	FEAT 4	FEAT 2	FEAT 14	FEAT 17	FEAT 14	
FEAT 14	FEAT 4	FEAT 19	FEAT 3	FEAT 3	FEAT 14	FEAT 15	
FEAT 4	FEAT 10	FEAT 2	FEAT 9	FEAT 4	FEAT 10	FEAT 3	
FEAT 10	FEAT 2	FEAT 9	FEAT 10	FEAT 11	FEAT 3	FEAT 2	
FEAT 19	FEAT 19	FEAT 10	FEAT 11	FEAT 9	FEAT 9	FEAT 9	
FEAT 9	FEAT 9	FEAT 11	FEAT 17	FEAT 2	FEAT 2	FEAT 19	
FEAT 11	FEAT 11	FEAT 17	FEAT 19	FEAT 19	FEAT 11	FEAT 11	NA
most sensitive					least sensitive		

Figure 18: Feature ranking of the Tridacna Crocea (TCR) data set. The NA indicates a negative change in total cost due to the Marxan error. A ranking of this feature is therefore not possible. The blue-marked feature 17 is an example for different positioning of features in the Marxan ranking if their shadow prices are not stable.



LP 0	LP -300	LP -600	LP -900	LP -5000	Marxan -5000	Marxan -7500	Marxan -10000
FEAT 21	FEAT 21	FEAT 21	FEAT 21	FEAT 21	FEAT 21	FEAT 21	FEAT 21
FEAT 26	FEAT 26	FEAT 26	FEAT 26	FEAT 14	FEAT 23	FEAT 23	FEAT 26
FEAT 23	FEAT 23	FEAT 23	FEAT 23	FEAT 26	FEAT 26	FEAT 26	FEAT 23
FEAT 11	FEAT 14	FEAT 14	FEAT 14	FEAT 23	FEAT 11	FEAT 14	FEAT 14
FEAT 14	FEAT 11	FEAT 24	FEAT 24	FEAT 18	FEAT 14	FEAT 18	FEAT 18
FEAT 12	FEAT 24	FEAT 18	FEAT 18	FEAT 25	FEAT 24	FEAT 11	FEAT 25
FEAT 24	FEAT 18	FEAT 25	FEAT 25	FEAT 11	FEAT 18	FEAT 24	FEAT 11
FEAT 18	FEAT 25	FEAT 11	FEAT 11	FEAT 13	FEAT 13	FEAT 25	FEAT 16
FEAT 25	FEAT 12	FEAT 12	FEAT 12	FEAT 16	FEAT 25	FEAT 15	FEAT 24
FEAT 13	FEAT 13	FEAT 19	FEAT 19	FEAT 19	FEAT 12	FEAT 12	FEAT 13
FEAT 20	FEAT 19	FEAT 13	FEAT 13	FEAT 24	FEAT 16	FEAT 16	FEAT 19
FEAT 19	FEAT 20	FEAT 20	FEAT 20	FEAT 15	FEAT 20	FEAT 13	FEAT 15
FEAT 16	FEAT 16	FEAT 16	FEAT 16	FEAT 12	FEAT 22	FEAT 19	FEAT 17
FEAT 15	FEAT 15	FEAT 15	FEAT 15	FEAT 17	FEAT 15	FEAT 17	FEAT 10
FEAT 17	FEAT 22	FEAT 22	FEAT 22	FEAT 10	FEAT 19	FEAT 20	FEAT 22
FEAT 22	FEAT 17	FEAT 17	FEAT 17	FEAT 20	0 FEAT 10	FEAT 10	FEAT 12
FEAT 10	FEAT 10	FEAT 10	FEAT 10	FEAT 22	0 FEAT 17	FEAT 22	FEAT 20
most sensitive					least sensitive		

Figure 19: Feature ranking of the Tasmania (TAS) data set. The zeros signify that the shadow prices of features 20 and 22 are 0. Therefore, they are both holding the same position within the ranking.

6 Discussion and Conclusion

In this thesis, we sought to develop a method which describes the influence of features on total cost of a reserve system with respect to changes in their targets. In this context, we aimed to create a feature ranking and find out how much targets can be manipulated so that total cost would remain the same or change drastically.

Since there is a variety of problem formulations available (see Chapter 2, we focused only on the *minimum set coverage problem* and its relaxation, the *land allocation problem*. In Chapter 3, we pointed out that large ILPs cannot be solved by direct resolution methods, such as branch and bound, in reasonable computation time. In this case, heuristic methods like simulated annealing can be used. The LP, however, can be solved quickly by applying the simplex algorithm.

For LPs, the impact of changes in the RHS (targets) on the objective function value (total cost) can be described by shadow prices. In Chapter 4, we presented two different shadow price definitions and proved their equivalence. We also found out that under degeneracy, the shadow price is not unique and we defined the two-sided 'true' shadow price. In the last part of this chapter, different approaches of conducting a sensitivity analysis for ILPs have been presented.

The results of the previously mentioned chapters have been used to create a feature ranking based on their influence on total conservation cost. The underlying methodology uses the shadow prices of the LP relaxation of the *minimum set coverage problem.* The obtained ranking is then validated by two other feature rankings, created with simulated annealing (Marxan) and - if possible - by applying branch and bound to a sequence of ILPs. Here, the aim was to find out if shadow prices can be used as an indicator for the sensitivity of the features of an ILP.

In Chapter 5, we applied this method to three real-world conservation prioritization problems. We observed that for stable shadow prices all three rankings showed similar results for the most and least sensitive features for all tested data sets.

If it is possible to solve an ILP relatively quick, the best way of creating a feature ranking is probably applying one of the methods presented in Chapter

4.2. Since this is not applicable for most of the problems considered in this thesis, we used the relaxation (an LP) instead, assuming that their optimal solutions are close to each other. But this assumption must be regarded with caution, as the following example, communicated by A. Kripfganz, will show.

Example 6.1: (*Distance between the optimal ILP and LP solution*)
Consider the following optimization problem with $k \in \mathbb{R}$.

$$\begin{array}{rlrl} & \max_{\mathbf{x}} x_1 + kx_2 & & \\ \text{s.t.} & x_1 \quad +(k+1)x_2 & \leq & k+1 \\ & \qquad\qquad\quad x_1 & \leq & \frac{1}{2}k \end{array}$$

Figures 20 and 21 show the graphical solutions for $k = 1$ and $k = 3$. In both figures, the point P_1 represents the integer and P_2 the continuous solution. It can be seen that for increasing values of k the LP solution recedes from the ILP solution.

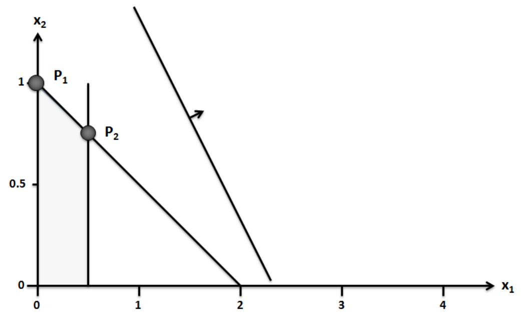


Figure 20: Graphical solution for $k = 1$. P_1 is the ILP and P_2 the LP optimum.

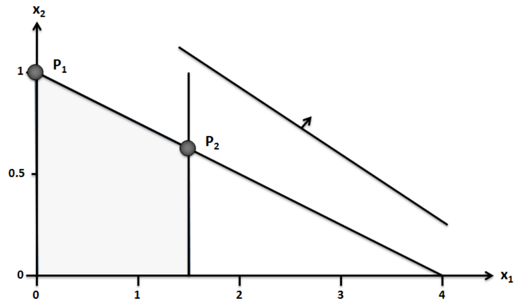


Figure 21: Graphical solution for $k = 3$. P_1 is the ILP and P_2 the LP optimum.

This example demonstrates that LP and ILP solutions of the same underlying optimization problem do not necessarily have to be close to each other. This means that in general, it is not possible to approximate an optimal ILP solution by its relaxation. However, this approach seems to work quite well in practice. This can be seen in Figures 12 and 13 for instance, which display the conservation maps of the optimal ILP and LP solution of the MPA data set. Many planning units chosen by both approaches are identical.

As we have seen in the TCR example, the quality of the shadow price feature ranking depends on its stability. A ranking is understood to be stable, if there is no major change in the ranking position of each dual, even though we are considering higher than marginal perturbations. The validity range of each shadow price can be obtained with the help of linear parametric optimization, the topic of which we covered in Chapter 4.1.2. It is thus recommendable, that a duals stability check (see Algorithm 5.2) is always conducted in order to validate the credibility of feature rankings based on shadow prices.

The subject of degeneracy and its effects on shadow prices also have an influence on the ranking. Since the conservation prioritization problems considered in this thesis are usually overdetermined, degeneracy is likely to occur. Yet, many problem solvers provide only one shadow price, even under degeneracy (e.g. LPSolve [4]). It would be necessary, to implement appropriate methods that calculate the true shadow prices and create rankings according to these. Nevertheless, we should first consider the question if the impact of degeneracy on the ranking is high enough to falsify it. In this context, the duals stability check might take away some of the uncertainties. In fact, for the ranking itself, it is only the position in relation to other features that matters and not the actual shadow price value.

Apart from the more technical aspects discussed above, the quality of the ranking is also highly dependent on the underlying data, since “data inputs into the problem are often imprecise” [18, p. 189]. Consider for example the cost vector. “Although only a single cost can be defined for each planning unit, this cost can be a composite of different measures, provided there is a defensible basis with which to combine them. Costs of the same currency can be combined (e.g. if both costs are monetary). Costs of a different currency can not sensible be combined without using arbitrary weightings (e.g. if one cost is monetary and one is social).” [11, p. 46]

Having this in mind, the interpretation of shadow prices in the context of conservation prioritization problems must be done carefully. The ranking is based on the respective feature's influence on total cost and in this regard, the shadow prices are not meant to be 'price tags' for the considered features. Additionally, it is recommendable to always examine several of the most sensitive features for target adjustments. Due to the fact that the ranking bases on approximations, the existence of 'the' most sensitive feature cannot be guaranteed. Furthermore, the most sensitive features also tend to be the rarest (e.g. species) and thus represent those which decision makers would least like to compromise on [23]. The reason for this is that their 'share' in the costs of a planning unit is relatively high, compared to more common features.

Consequently, one can draw the conclusion that shadow prices can indeed be used as an indicator for the influence of a feature on the total cost of a reserve system. A respective feature ranking is valid for both, ILPs like the *minimum set coverage problem* and LPs, such as the *land allocation problem*. For the latter, shadow prices even provide information about actual changes in total cost, due to changes in the targets. However, they must be handled carefully, especially regarding their interpretation as 'prices'.

The results of Chapter 5 validate this conclusion, though it would be interesting to test the suggested methodology on other data sets with different data structure. Additionally, further research is needed in order to expand the method and make it applicable to more complex problem formulations. Based on an increasing computational capacity, it might also be worthy to develop better implementations of the ILP sensitivity analysis approaches, in order to create more accurate ILP feature rankings.

An outlook on how the results of this thesis can be applied and some more suggestions for further research will be given in the next chapter.

7 Outlook

In the long term, it would be desirable to include feature rankings like the one presented in this thesis into Marxan in order to provide an additional tool for conservation planners, supporting them in compromising on the target setting of conservation features. Since LPSolve is freely available and can also be called from the R software environment, the solver is compatible with Marxan and could easily be integrated into it.

As a 'by-product' of this thesis, we observed that for the considered data sets, many of the selected planning units by the LP solver and Marxan were identical. For a run with a 30% target for each feature of the TAS data set Marxan, selected for example 345 planning units for the ILP solution and LPSolve 319 planning units for the LP solution, 230 of these planning units were identical (see respective file on the CD). This comparison does not take into account variable values. However, most variables of the LP solution had the value 1. The reason for this is rather simple. In every iteration, the simplex algorithm selects as high a share of the 'cheapest' site containing the desired feature as is needed to meet its target. If the target cannot be met by protecting only one site, it selects the full planning unit and skips to the next cheapest site. Thus, many of the variables take values at their upper bound.

This fact can be used to improve the species penalty factor (SPF), Marxan uses in order to make sure that most or all targets are met. The higher the SPF, the more likely it is that the obtained solution meets all targets. Here, we used by default a SPF of 10. More information about the SPF can be found in [11].

Another idea which might improve Marxan results and uses the above mentioned observation, is to provide simulated annealing with an optimal LP solution as a starting point and run the algorithm with a low temperature. If the ILP solution is close to the LP solution, Marxan would then be able to detect it relatively fast.

The most recent accomplishments of the research referred to Marxan and the ideas discussed in this chapter can be found on Matthew Watt's GitHub page [27].

A Branch & Bound Flow Chart

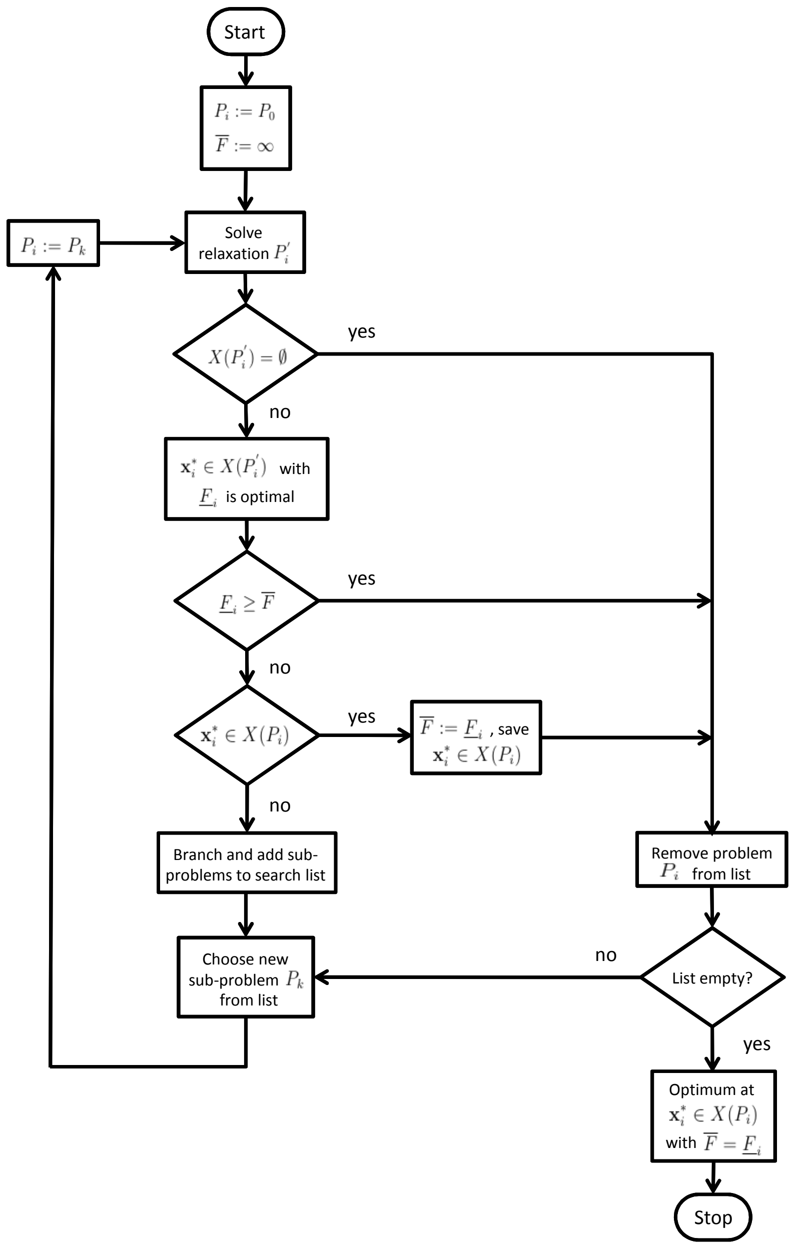

Start
$P_i := P_0$
$\overline{F} := \infty$
$P_i := P_k$
Solve relaxation P_i'
$X(P_i') = \emptyset$
yes
no
$\mathbf{x}_i^* \in X(P_i')$ with $\underline{F}_i$ is optimal
$\underline{F}_i \geq \overline{F}$
yes
no
$\mathbf{x}_i^* \in X(P_i)$
yes
$\overline{F} := \underline{F}_i$, save $\mathbf{x}_i^* \in X(P_i)$
no
Branch and add sub-problems to search list
Remove problem P_i from list
Choose new sub-problem P_k from list
no
List empty?
yes
Optimum at $\mathbf{x}_i^* \in X(P_i)$ with $\overline{F} = \underline{F}_i$
Stop

Figure 22: Modified flow chart of the branch and bound algorithm for minimization problems from Domschke and Drexl 2005 [9].

B Summary of the Dual Bounded Simplex Algorithm

Consider the *minimum set coverage problem*

$$\min\{\mathbf{c}^T\mathbf{x} \mid A\mathbf{x} \geq \mathbf{b}, 0 \leq \mathbf{x} \leq \mathbf{u}\}.$$

Step 1 Initial problem formulation

Convert the given optimization problem to the form

$$-\max\{-\mathbf{c}^T\mathbf{x} \mid -A\mathbf{x} \leq -\mathbf{b}, 0 \leq \mathbf{x} \leq \mathbf{u}\}.$$

Step 2 Slack variables

Add slack variables to the constraints and the upper bound relations, so that the optimization problem takes the form

$$-\max\{-\mathbf{c}^T\mathbf{x} \mid -A\mathbf{x} + I\tilde{\mathbf{x}} = -\mathbf{b}, 0 \leq \mathbf{x} \leq \mathbf{u}, \tilde{\mathbf{x}} \geq 0\}$$

and $\mathbf{x} + \mathbf{x}' = \mathbf{u}$.

Set up the initial dual simplex tableau

	$\mathbf{x}$	$\tilde{\mathbf{x}}$	
z	$-\mathbf{c}$	$\tilde{\mathbf{c}}$	0
$\tilde{\mathbf{x}}$	$-A$	I	$\mathbf{b}$

Step 3 Optimality check

If

(i) $-b_j \geq 0 \;\; \forall j$,

(ii) $\hat{c}_s \leq 0 \;\; \forall s \in \{1, ..., n+m\}$ and

(iii) $0 \leq x_i \leq u_i \;\; \forall i$

then, the current tableau is optimal, i.e. go to step 8. If at least one of the conditions is not fulfilled, got to step 4.

Step 4 Pivot element

Choose the row with the most negative b_j, say the k-th. The pivot element $\hat{a}_{kl}$ is the one that fulfils

$$\min\left\{\frac{z-\hat{c}_s}{-\hat{a}_{ks}}\right\} \forall s \in \{1,...,n+m\}, \hat{a}_{ks} < 0.$$

Step 5 Pivot step

Calculate the new values of the pivot row

$$\hat{a}'_{ks} := \frac{\hat{a}_{ks}}{\hat{a}_{kl}} \quad \text{and} \quad -b'_k := \frac{-b_k}{\hat{a}_{kl}}$$

and then those of the remaining rows

$$\hat{a}'_{js} := \hat{a}_{js} + \hat{a}'_{ks} \cdot (-\hat{a}_{jl}) \quad \text{and} \quad -b'_j := -b_j + (-b'_k) \cdot (-\hat{a}_{jl}) \; \forall j, j \neq k$$

so that $\hat{a}'_{kl} = 1$ and all other elements of the pivot column are zero.

Step 6 Boundary check

If all variables are within their bounds, go to step 3. If at least one variable exceeds its upper bound, continue with step 7.

Step 7 Variable substitution

If a variable of the r-th row of the basis, say x_p, exceeds its upper bound,

(i) change all signs of the r-th row, except of the factor $\hat{a}_{rp} = 1$,

(ii) replace the value of x_p by $x'_p = u_p - x_p$ in the RHS,

(iii) replace the lable of x_p by x'_p,

(iv) calculate $z' = z + c_p u_p$ and

(v) change the sign of c_p.

Go to step 3.

Step 8 Optimal solution

In the final optimal tableau, the non-basis variables are zero and the basis variables take the values of the RHS. Note that all complementary variables have to be re-substituted by

$$x_i = u_i - x'_i.$$

Stop.

Acknowledgement

First and foremost I have to thank my supervisors Ms. Anita Kripfganz and Mr. Hugh P. Possingham who enabled me to write this thesis partly in Germany and partly in Australia. I am very grateful for the opportunity Mr. Possingham gave me when he invited me to Brisbane. Working in his lab at the University of Queensland was an extraordinary experience for me. Ms. Kripfganz's profound knowledge about mathematical optimization was an immense benefit. I appreciate her patience explaining complex matters to me and her assistance with the proofs given in this thesis. I also thank Matthew Watts for running all the data sets in Marxan and my friends Hedi and Christian for reviewing the thesis. Most importantly, I would like to show my gratitude to my grandmother who is always supporting me in any possible way.

References

[1] Akgül, M 1984, 'A Note on Shadow Prices in Linear Programming', *The Journal of the Operational Research Society*, vol. 35, no. 5, pp. 425–431.

[2] Aucamp, DC, Steinberg, DI 1982, 'The Computation of Shadow Prices in Linear Programming', *The Journal of the Operational Research Society*, vol. 33, no. 6, pp. 557–565.

[3] Avriel, M 2003, *Nonlinear Programming: Analysis and Methods,* Dover Publications, New York.

[4] Berkelaar, M, Eikland, K, Notebaert, P 2011, *lp_solve (version 5.5.2.0)*, <http://lpsolve.sourceforge.net/5.5>.

[5] Bertsimas, D, Tsitsiklis, JN 1997, *Introduction to Linear Optimization,* Athena Scientific, Belmont, MA.

[6] Burgman, MA, Breininger, DR, Duncan, BW, Ferson, S 2001, 'Setting Reliability Bounds on Habitat Suitability Indices', *Ecological Applications*, vol. 11, no. 1, pp. 70–78.

[7] Burkhard, RE, Zimmermann, UT 2012, *Einführung in die Mathematische Optimierung,* Springer-Verlag, Berlin Heidelberg.

[8] Cocks, KD, Baird, IA 1989, 'Using Mathematical Programming to Address the Multiple Reserve Selection Problem: An Example from the Eyre Peninsula, South Australia', *Biological Conservation*, vol. 49, no. 2, pp. 113—130.

[9] Domschke, W, Drexl, A 2005, *Einführung in Operations Research,* Springer-Verlag, Berlin Heidelberg.

[10] Gal, T 1986, 'Shadow Prices and Sensitivity Analysis in Linear Programming under Degeneracy', *Operations-Research-Spektrum*, vol. 8, no. 2, pp. 59–71.

[11] Game, ET, Grantham, HS 2008, *Marxan User Manual: For Marxan version 1.8.10.*, The University of Queensland, Brisbane, and Pacific Marine Analysis and Research Association, Vancouver.

[12] Guzelsoy, M 2010, 'Dual Methods in Mixed Integer Linear Programming', PhD thesis, Lehigh University, Bethlehem, PA.

[13] Guzelsoy, M, Ralphs, TK 2010, 'Integer Programming Duality', in Cochran et al. (eds.), *Wiley Encyclopedia of Operations Research and Management Science*, John Wiley & Sons, Inc., Hoboken, NJ.

[14] Ho, JK 2000, 'Computing True Shadow Prices in Linear Programming', *Informatica*, vol. 11, no. 4, pp. 421–434.

[15] Kim, S, Cho, S 1988, 'A shadow price in integer programming for management decision', *European Journal of Operational Research*, vol. 37, no. 3, pp. 328–335.

[16] Knolmayer, G 1984, 'The effects of degeneracy on cost-coefficient ranges and an algorithm to resolve interpretation problems', *Decision Sciences*, vol. 15, no. 1, pp. 14–21.

[17] Koopmans, TC 1976, 'Concepts of optimality and their uses', *Mathematical Programming*, vol. 11, no. 1, pp. 212–228.

[18] Moilanen, A, Wilson, KA, Possingham, HP (eds.) 2009, *Spatial Conservation Prioritization: Quantitative Methods & Computational Tools,* Oxford University Press, New York. Foreword by Ilkka Hanski.

[19] Nickel, S, Stein, O, Waldmann, KH 2011, *Operations Research,* Springer-Verlag, Berlin Heidelberg.

[20] Possingham, HP, Ball, I, Andelman, S 2000, 'Mathematical methods for identifying representative reserve networks', in Ferson, S, Burgman, M (eds.), *Quantitative methods for conservation biology*, Springer-Verlag, New York, pp. 291–305.

[21] Salt, D 2010, *Decision Point - Special Marxan Issue,* Applied Environmental Decision Analysis hub - AEDA, Brisbane.

[22] Shaffer, ML 1981, 'Minimum Population Sizes for Species Conservation', *BioScience*, vol. 31, no. 2, pp. 131–134.

[23] Soulé, ME, Simberloff, D 1986, 'What do genetics and ecology tell us about the design of nature reserves?', *Biological Conservation*, vol. 35, no. 1, pp. 19–40.

[24] The University of Queensland 2011, *Draft Marxan Tutorial - Module 2: Theory behind Marxan*, viewed 16 December 2014, <http://www.uq.edu.au/marxan/tutorial/module2.html>.

[25] TU Clausthal 2011, *Simulated Annealing*, viewed 04 December 2014, <http://www.iasor.tu-clausthal.de/Arbeitsgruppen/Stochastische-Optimierung/forschung/simulated-annealing/>.

[26] Wagner, HM 1958, 'The dual simplex algorithm for bounded variables', *Naval Research Logistics Quarterly*, vol. 50, no. 3, pp. 257–261.

[27] Watts, M 2014, *Marxan GitHub repository*, <https://github.com/mattwatts/>.

[28] Watts, M, Possingham, HP n.d., *R Shiny Server apps*, viewed 23 January 2015, <http://marxan.net/shinyapps.html>.

[29] Watts, ME, Ball, IR, Stewart, RS, Klein, CJ, Wilson, K, Steinback, C, Lourival, R, Kircher, L, Possingham, HP 2009, 'Marxan with Zones: software for optimal conservation based land-and sea-use zoning', *Environmental Modelling & Software*, vol. 24, no. 12, pp. 1513–1521.

Declaration of Authorship

I declare that the work presented here is, to the best of my knowledge and belief, original and the result of my own investigations, except as acknowledged, and has not been submitted, either in part or whole, for a degree at this or any other university. Formulations and ideas taken from other sources are cited as such. This work has not been published.

Leipzig, 02.03.2015

Location, Date

[illegible]

Signature