Delft University of Technology
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft Institute of Applied Mathematics

Identifying the most relevant cost-functions for radiotherapy treatment planning with multicriteria optimisation

(Dutch title: De meest relevante kostfuncties voor radiotherapie behandelplannen identificeren met multicriteria optimalisatie)

Report for the
Delft Institute of Applied Mathematics
to obtain the degree of

BACHELOR OF SCIENCE
in
APPLIED MATHEMATICS

by Kelly Vos
Delft, The Netherlands
June 2019

Abstract

Cancer is a disease that one of every three people will get in The Netherlands. One of the treatment methods for this disease is radiotherapy. Approximately half of all cancer patients will get radiotherapy at some point of their treatment. During radiotherapy cancer cells are destroyed with ionizing radiation, but healthy cells get destroyed too. When a patient gets treated with radiotherapy, the goal is to find a treatment plan which will destroy all of the cancer cells and as few healthy cells as possible. To reach this goal we want to make a unique treatment plan for every patient, because every patient is anatomical unique.

We use a wish-list to generate this unique optimal treatment plan. This wish-list contains all of the demands of the physician. All of the demands can be written into cost-functions. We will use inverse multicriteria optimisation to find the most relevant cost-functions for every organ and the tumour (planning target volume (PTV)). The relevance of a cost-function can be obtained by determining the weight of a cost-function. We start with a non-linear problem and we use the Karush-Kuhn-Tucker conditions. We did not receive the desired solutions. Afterwards, we tried to find the optimal weights for a linear problem by writing it in the form of an absolute duality gap minimization problem. This gave the results we were hoping for.

Contents

PREFACE

This report is the result of my Bachelor project for the Delft University of Technology. For this project I did an internship at the Erasmus Medical Center in Rotterdam. This report contains general information about radiotherapy and automated treatment planning. Furthermore, it contains the research I did focused on mathematical methods to optimise treatment plans for radiotherapy.

During my project I was supported by Sebastiaan Breedveld at the Erasmus Medical Center and Marleen Keijzer at the Delft University of Technology. While doing this project I learned a lot about radiotherapy, optimisation and doing research. I want to thank my supervisors for their help and support during this project.

Kelly Vos
Delft, June 2019

1 INTRODUCTION

Cancer is a disease that one of every three people will get in The Netherlands. For this disease we have different treatment methods. Examples are chemotherapy, surgery and radiotherapy. Approximately half of all cancer patients will get radiotherapy at some point of their treatment. This thesis will focus only on radiotherapy.

During radiotherapy cancer cells are destroyed with ionizing radiation, but healthy cells get destroyed too. When a patient gets treated with radiotherapy, the goal is to find a treatment plan which will destroy all of the cancer cells and as few healthy cells as possible. To reach this goal we want to make a unique treatment plan for every patient, because every patient is anatomical unique.

We use a wish-list to generate this unique optimal treatment plan. This wish-list contains all of the demands of the physician. For example, the maximum dose in a certain organ. All of the demands can be written into cost-functions.

We will use inverse multicriteria optimisation to find the most relevant cost-functions for every organ and the PTV. We want to find the most relevant cost-functions, because when a patient comes in for treatment we want to generate an optimal treatment plan quickly. The fewer cost-functions we have to test, the less time it takes to make the treatment plan. The relevance of a cost-function can be obtained by determining the weight of a cost-function.

Chapter 2 contains some general information about radiotherapy, treatment plans and the Cyberknife, a treatment unit. In this chapter there is also an explanation of a method to determine if a treatment plan is good or bad. Chapter 3 begins with the the basic principles of optimisation and an explanation of multicritaria optimisation. Followed by an explanation about the link between optimisation and treatment planning. This chapter ends with information about, and an example of the wish-list. In Chapter 4 inverse multicriteria optimisation with the Karush-Kuhn-Tucker conditions is explained for a non-linear problem and it is made insightful with an example. This chapter ends with the implementation of this non-linear problem in MATLAB. Chapter 5 contains the results of the (inverse) multicriteria optimisation. In chapter 6 a wish-list for a linear problem is formulated and we write the problem such that it is in the form of an absolute duality gap minimization problem. Chapter 7 contains the results of the (inverse) optimisation of the linear problem. Chapter 8 gives an overview of the most important results. Finally, in Chapter 9 some comments and recommendations are made.

2
Radiotherapy

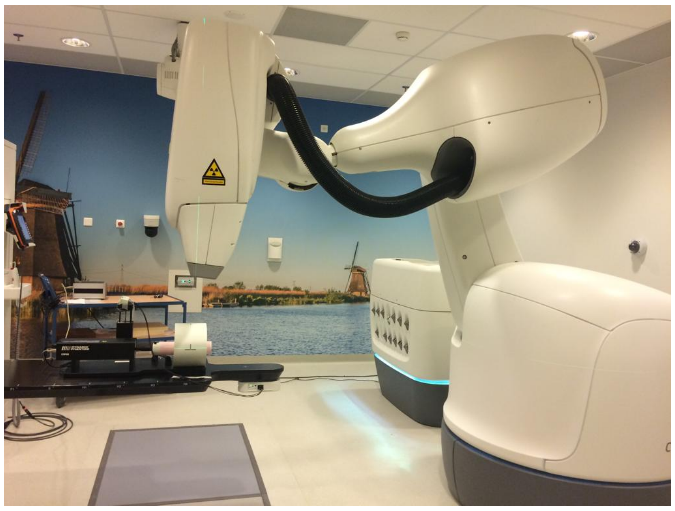
In The Netherlands there is a one in three chance that someone will get cancer in their lifetime[1]. This means that every year one hundred and sixty thousand people get diagnosed with cancer in The Netherlands. There are different ways to treat cancer, mainly by surgery, chemotherapy and radiotherapy. Often the treatment of a patient is a combination of multiple treatments. Radiotherapy is used in approximately half of all treatments.

Radiotherapy is a method to treat cancer with ionizing radiation. The goal of radiotherapy is to destroy as many malignent cells (tumour cells) as possible while preserving as many healthy cells as possible. There are two ways to treat a patient with radiotherapy. Firstly, it is possible to temporary place radioactive sources inside the patient. This is called brachytherapy (BT). A second method to treat a patient with radiotherapy is external beam radiation therapy (EBRT). This means that the patient is irradiated from the outside of their body by a device as shown in Figure 2.1. This report focuses on EBRT.

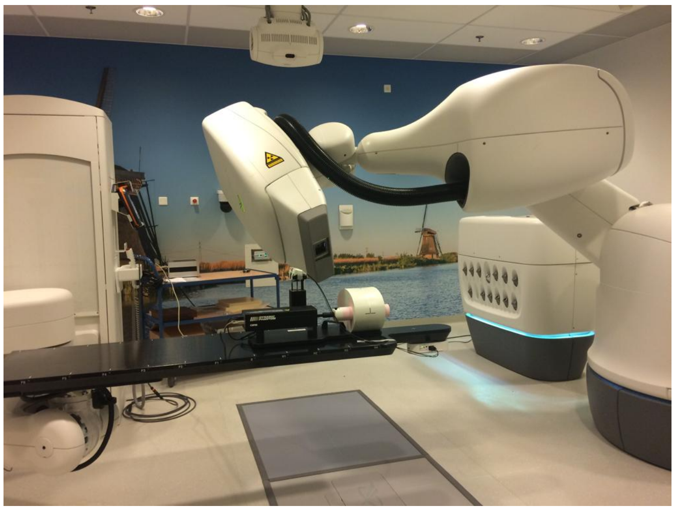
2.1. Treatment plan

Before the patient is treated with radiotherapy, a treatment plan needs to be generated. This plan contains the settings for the treatment, such that the machine is able to deliver the desired dose in the patient's organs and tumour (planning target volume (PTV)). The treatment plan for each patient is unique because of two reasons: every human is anatomically unique and every tumour is different. Therefore, in order to define the correct treatment plan, the anatomy of the patient has to be clearly defined. The exact location of the tumour and organs at risk (OAR) will define the distribution of the dose.

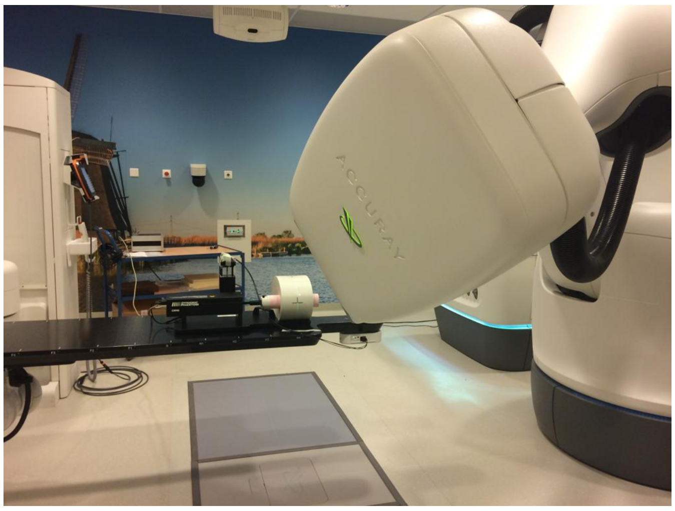
[1] Incidence and survival rates: Dutch Cancer Registry, managed by IKNL © February 2018 (provisional incidence rates from 2017) Mortality rates: CBS provisional figures from 2017



(a)



(b)



(c)

Figure 2.1: Different positions of the Cyberknife arm

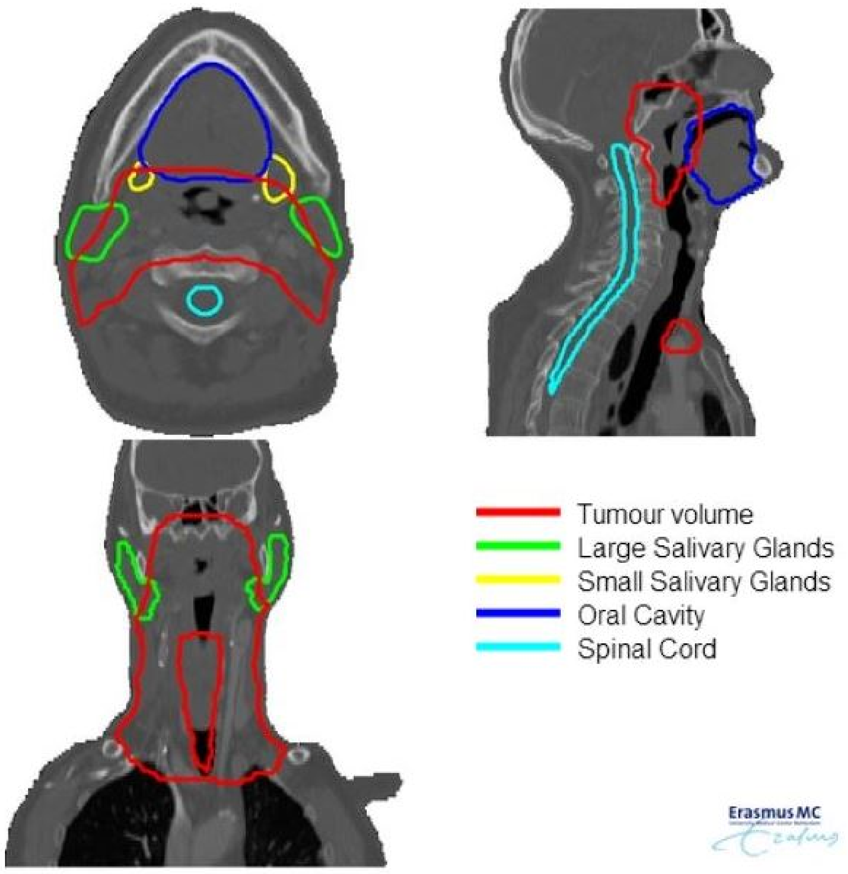


Figure 2.2: Example of a CT-scan of the head-and-neck area of which the relevant parts are outlined ([1])

This information about the location of the tumour and OAR is obtained by making a computed tomography scan (CT-scan)[2] of the patient. The CT-scan (an example is given in Figure 2.2) gives the location and size of the different OAR and the PTV. For modelling, the OAR and the PTV are discretized into small imaginary volumes called voxels. A wish-list for the treatment plan is then formulated out of the CT-scan, the voxels and the maximum dose that each OAR can handle. The concept of a wish-list will be explained further in Section 3.4. The wish-list represents a multicriteria optimisation problem. When this problem is solved we get a treatment plan.

[2]Sometimes another scan is used to identify the location of the tumour. Some alternatieves are: a magnetic resonance imaging (MRI), a positron-emission tomography scan (PET-scan) or a single photon emission computed tomography scan (SPECT-scan)

When the treatment plan is finished, the physician will inspect the plan visually. If the physician thinks the treatment plan is fine, the treatment plan is delivered in different fractions. The treatment is spread out over different days, sometimes even up to forty days. Each day the patient receives a fraction of the plan. The time between different radiation sessions gives the body a chance to recover from the radiation. This reduces the damage to the OAR.

2.2. DOSE-VOLUME HISTOGRAM

When generating a treatment plan we want to get a clear vision of the effects on the OAR and the PTV of this treatment plan. One way to analyse the effects is by inspecting the 3D dose distribution. Another option is by generating a dose-volume histogram (DVH). This is a histogram that shows a 2D representation of the 3D dose distribution of a given treatment plan. In a DVH it is possible to see how much dose (in Gy) a certain percentage of an OAR or PTV gets. An example of a DVH is shown in Figure 2.3.

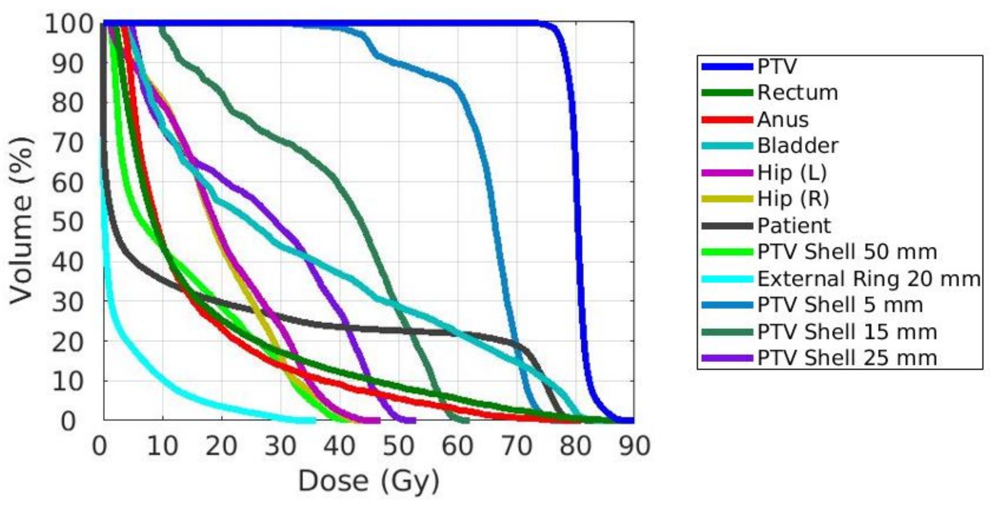


Figure 2.3: Example of a DVH

Most of the OAR are well known but there are a few OAR that need some explanation. Firstly, there are three PTV shells, all with a different distance. These shells are zones around the PTV. The PTV Shell 5 mm, for example, is the zone from the PTV to 5 mm outside the PTV. The shells are used to obtain a big difference between the dose delivered in the PTV and directly next to the PTV. The goal is to reduce the dose delivered directly next to the PTV. Secondly, the label 'patient' means unspecified tissue, such as fat cells and muscles. Finally, the External Ring 20 mm. This OAR is implemented to minimise the dose absorbed below the surface of the skin ([2]).

With a DVH we gain insight on the quality of a treatment plan. For an OAR, the goal is to have a low dose in most of its total volume. For a PTV, the goal is to have a high dose in a big percentage of its total volume. We are also able to compare the influence of different treatment plans on the OAR and the PTV.

2.3. CYBERKNIFE

There are different types of treatment devices. One example is the Cyberknife, a linear accelerator which is used for the treatments. The patient lies down on a table and a robotic arm moves around the patient to deliver the radiation according to the treatment plan. The Cyberknife moves around the patient so it can irradiate the patient from different directions. Different positions of the robotic arm are shown in Figure 2.1. The Cyberknife irradiates subsequently from different positions. Each beam may have a different intensity. These intensities are determined in the treatment plan.

REFERENCES

[1] S. Breedveld, *Radiotherapy,* (2013), [Online; accessed June 16, 2019].

[2] R. Van Haveren, *Lexicographic Reference Point Method for Automatic Treatment Planning in Radiation Therapy*, Master's thesis, Delft University of technology, the Netherlands (2014).

3
Multicriteria optimisation

3.1. Basic optimisation

The idea of optimisation is to find the "best solution" for a problem. In this project we will only use minimisation. The goal is to minimize a convex[1] function $f(x)$, which is called an objective cost-function. So we want to find the values $x \in \mathbb{R}^N$, such that $f(x)$ is as small as possible. The values of the vector x are called the decision variables.

There are restrictions that the solution has to meet. These restrictions are given in the form of constraints. These are m cost-functions $g_j(x)$, each with an upper bound $b_j \in \mathbb{R}$. All of the above gives the following optimisation problem:

$$\begin{aligned} \min_x \quad & f(x) \\ \text{subject to} \quad & g_1(x) \le b_1 \\ & \vdots \\ & g_m(x) \le b_m \end{aligned} \tag{3.1}$$

3.2. Multicriteria optimisation

An optimisation problem is called a multicriteria problem if the problem contains more than one objective function. Objective functions can be unevenly weighted. This is included in the problem by multiplying each objective cost-function f_i with a weight $w_i \in \mathbb{R}^n$. This gives the following optimisation problem:

$$\begin{aligned} \min_x \quad & \sum_{i=1}^{n} w_i f_i(x) \\ \text{subject to} \quad & g_1(x) \le b_1 \\ & \vdots \\ & g_m(x) \le b_m \end{aligned} \tag{3.2}$$

[1]A function $f : \mathbb{R}^n \to \mathbb{R}^n$ is called convex if: $\forall x, y \in \mathbb{R}^n$ and $t \in (0,1) : f(tx + (1-t)y) \le tf(x) + (1-t)f(y)$

The goal of this problem is to find an optimal solution. We want to find a solution vector x such that $\sum_{i=1}^{n} w_i f_i(x)$ is as small as possible while the constraints are met. In this report we use the interior-point solver developed in the Erasmus MC ([1]).

3.3. Treatment planning and optimisation

When optimisation is used for treatment planning, the decision variables x contain the information about a specific beamlet of the Cyberknife. Examples of such information are the intensity and angle of the beamlet. The objective cost-functions $f_1(x),\ldots,f_n(x)$ quantify the influence of a solution x. There can be multiple cost-functions working on the same OAR or PTV, with different priorities. For example a cost-function can have the goal to maximize the minimum dose of a tumour. Another cost-function has for example the goal to minimize the mean dose in an OAR.

The constraint cost-functions $g_1,\ldots,g_m$ in combination with their limits $b_1,\ldots,b_m$ give the restrictions for different OAR and the PTV. The wishlist contains all the information about the cost-functions: the name of the OAR or PTV, priority, type and goal. See an example of a wishlist in Table 3.1.

There are five different types of cost-functions. All the cost-functions except the quadratic cost-functions use $d(x) = Ax + h$, with $d(x)$ the dose in a voxel of a particular OAR or PTV in Gy (J/kg).

A is the data matrix, in radiotherapy this is the pencil-beam dose matrix. This matrix A can be different for each objective- or constraint cost-function, but for every OAR or PTV the number of columns is the same: the number of decision variables x. The number of rows can differ, because that equals the number of voxels (k) of the OAR or PTV. The vector h is usually zero, only in special cases h has another value, for example if the patient already received a certain dose (for example due to a previouos treatment). The Erasmus-iCycle solver[2] supports the following commonly used cost-functions:

1. *Linear:* there are three different options for cost-functions of this type. The first option is the pointwise minimum $f(x) = \min(d(x))$. The second option is the pointwise maximum $f(x) = \max(d(x))$. The last option is that the cost-function computes the mean $f(x) = \frac{1}{k}\sum_{i=1}^{k} d_i(x)$.

2. *Quadratic*: this type of cost-function is computed by $f(x) = \frac{1}{2}x^T Cx + qTx + r$

[2]The method used in the Erasmus MC to generate a treatment plan for cancer patients, using the wish-list and the CT-scan of a patient's anatomy

3. *gEUD* (generalised equivalent uniform dose): this type of cost-function is similar to the generalized mean with the formula $f(x) = \left(\frac{1}{k}\sum_{i=1}^{k} d_i(x)^a\right)^{\frac{1}{a}}$. The parameter a is given. The cost-function is convex if $a \geq 1$, then the function can be minimized. If $a \leq -1$ then the cost-function is concave and needs to be maximized.

4. *LTCP*: this type of cost-function is called the logarithmic tumour control probability and is computed as $f(x) = \frac{1}{k}\sum_{i=1}^{k} e^{\alpha(d_i(x)-d^p)}$. The parameters α and d^p are given. This type of cost-function is only used for the PTV.

5. *DVH*: we do not use this type of cost-function in this report.

6. *Chain*: the chain function is used to include formulations which contain multiple standard cost-functions. There are two different options for cost-functions of this type. Both kinds of cost-functions are chain functions. A chain function is a function that uses the outcome of another function of the problem.

   - The first option is given by two elements (a, i), where a is a scalar and i the index of an other objective of the problem. In this case the cost-function is computed as $f(x) = a \cdot g_i(x)$ and if it has multiple rows as $f(x) = a_1 g_{i_1}(x) + a_2 g_{i_2}(x) + \ldots$.
   - The second option is given by three elements (a, i, j). In this case a and i are defined the same as before and j gives an index number. This option results in a constraint $a \cdot g_i(x) \leq x_j$.

3

3.4. WISH-LIST

An approach for generating a treatment plan is using a wish-list. This wish-list contains the goals for the treatment. Basically a wish-list contains everything a physician wants for a treatment. An example could be to set the minimum dose to the PTV (tumour) or the maximum dose on OAR. There are also practical considerations. For example, it is more important to spare nerves than unspecified tissue. The treatment plan needs to be deliverable as well. For example, it is not possible to give a really high dose on one spot and right next to it no dose at all. An example of a wish-list for prostate cancer is given in Table 3.1. In this example the goal is to minimize the objectives. The limit of the constraints is a maximum dose the OAR or PTV should receive.

The wish-list is the same for every patient within the same treatment protocol. The resulting treatment plans differ, because the data matrix A is different for every patient. This is in its turn a result of the anatomical uniqueness of every patient. Each patient has different voxels and a different number of decision variables. While for one patient the data matrix A has the same number of columns, the number of columns can be different for an other patient. This will also have consequences for the weights for the cost-functions for each patient.

Table 3.1: Example of a wish-list

Objectives

Priority	Volume	Type	Goal (Gy)
1	Rectum	3 ($a = 12$)	30
2	Rectum	3 ($a = 8$)	20
3	Rectum	1 (mean)	10
4	External Ring 20 mm	1 (min)	31.20
5	PTV Shell 5 mm	1 (min)	72.54
6	Anus	1 (mean)	10
7	PTV Shell 15 mm	1 (min)	54.60
8	PTV Shell 25 mm	1 (min)	39
9	Bladder	1 (mean)	40
10	Hip (L)	1 (mean)	20
10	Hip (R)	1 (mean)	20
11	Unspecified tissue	1 (mean)	100

Constraints

Volume	Type	Limit (Gy)
PTV	1 (min)	81.12
PTV Shell 50 mm	1 (min)	39
PTV	4 ($\alpha = 0.80$ and $d^p = 78$)	0.50
Rectum	1 (min)	78
Hip (L)	1 (min)	40
Hip (R)	1 (min)	40
Bladder	1 (min)	78
Unspecified tissue	1 (min)	81.12
Anus	1 (min)	78

REFERENCES

[1] S. Breedveld, B. van den Berg, and B. Heijmen, *An interior-point implementation developed and tuned for radiation therapy treatment planning,* Computational Optimization and Applications **68**, 209 (2017).

4 INVERSE MULTICRITERIA OPTIMISATION

For multicriteria optimisation, the weights w_i and the costfunctions $f_i(x)$ and $g_j(x)$ are fixed and we want to find an optimal solution x. For inverse multicriteria optimisation (IMO) we have the opposite: we have the optimal solution x and cost-functions. The goal is to find the optimal weights w_i. We start with Problem 4.1:

$$\begin{aligned}
\min_{x} \quad & \sum_{i=1}^{n} w_i f_i(x) \\
\text{subject to} \quad & g_1(x) \leq b_1 \\
& \vdots \\
& g_m(x) \leq b_m
\end{aligned} \tag{4.1}$$

We are going to rewrite this problem to a problem whit a fixed solution x (calculated with multicriteria optimisation as described in Section 3.2). Our goal is to refind the optimal weights $\widehat{w}_i$ for this problem.

4.1. INVERSE MULTICRITERIA OPTIMISATION

To find these weights, we first look at the Langrangian of Problem 4.1:

$$\mathscr{L}(x,y) = \sum_{i=1}^{n} w_i f_i(x) + \sum_{j=1}^{m} y_j(g_j(x) - b_j) \tag{4.2}$$

Equation 4.2 can be written in vector form as:

$$\mathscr{L}(x,y) = w^T f(x) + y^T(g(x) - b) \tag{4.3}$$

Here y is the vector of Lagrange multipliers. The Lagrangian is convex in x and concave in y, because Problem 4.1 is convex. A mathematical solver is used ([1]) to find an optimal solution $(\hat{x}, \hat{y})$ that minimises Problem 4.1. For such a minimum the so-called Karush–Kuhn–Tucker conditions hold, which are:

$$w^T \nabla f(\hat{x}) + \hat{y}^T \nabla g(\hat{x}) = 0 \tag{4.4}$$

$$g(\hat{x}) \leq b \tag{4.5}$$

$$\hat{y}^T (g(\hat{x}) - b) = 0 \tag{4.6}$$

Each of the conditions has a different function. Equation 4.4 ensures that the minimum of the Lagrangian 4.3 is attained. Condition 4.5 ensures that all of the constraints are satisfied. The complementary slackness condition (Equation 4.6) ensures that either the constraint is active (and thus $g > 0$), or inactive and $g = 0$.

For inverse optimization, we want to find a solution for w given an $\hat{x}$, that matches these conditions as well as possible. Therefore, we write $w^T \nabla f(\hat{x}) + \hat{y}^T \nabla g(\hat{x}) = \epsilon$ and $\hat{y}^T \nabla(g(\hat{x}) - b) = \gamma$ as conditions and we state that the goal is to minimise ϵ and γ by defining the function $\phi(\epsilon, \gamma) = ||\epsilon||_2^2 + ||\gamma||_2^2$ ([2]). This ensures that $w^T \nabla f(\hat{x}) + \hat{y}^T \nabla g(\hat{x})$ and $\hat{y}^T \nabla(g(x) - b)$ are as close to zero as possible. Secondly, to exclude the trivial solution, $(\hat{w} = 0)$ we set the condition: $\sum_{i=1}^{n} w_i = 1$. Finally, to ensure that none of the weights are negative, the conditions $w_i \geq 0$ for $i \in \{1, \ldots, n\}$ and $\hat{y}_i \geq 0$ for $j \in \{1, \ldots, m\}$ are introduced. This gives the following inverse optimisation problem:

$$\begin{array}{ll} \min\limits_{\epsilon,\gamma} & \phi(\epsilon, \gamma) = ||\epsilon||_2^2 + ||\gamma||_2^2 \\ \text{subject to} & w^T \nabla f(\hat{x}) + y^T \nabla g(\hat{x}) = \epsilon \\ & y^T (g(\hat{x}) - b) = \gamma \\ & \sum\limits_{i=1}^{n} w_i = 1 \\ & w_i \geq 0 \text{ for } i \in \{1, \ldots, n\} \\ & y_i \geq 0 \text{ for } j \in \{1, \ldots, m\} \end{array} \tag{4.7}$$

In Problem 4.7 we write y instead of $\hat{y}$, because we assume that we do not know $\hat{y}$ (we do know $\hat{x}$). Here $f(\hat{x})$ and $g(\hat{x})$ are the same as in Problem 4.1 and $\hat{x}$ is the optimal solution for the weights w. Then the inverse optimisation of Problem 4.7, using $\hat{x}$, should return the optimal weights $\hat{w}$. Because of the convexity of the problem w (the original weigths) should be equal to the new optimal weights $\hat{w}$.

4.2. EXAMPLE

To further explain the methods described in Sections 3.2 and 4.1, we give a basic analytic example. Let $f_1(x) = x^2$ and $f_2(x) = e^x$ (both convex) be objective cost-functions, both with weights 0.5, and assume there are no constraint cost-functions $g(x)$. This results in:

$$
\begin{aligned}
w &= \begin{bmatrix}0.5 & 0.5\end{bmatrix} && (4.8)\\
f(x) &= \begin{bmatrix}x^2 & e^x\end{bmatrix} \Longrightarrow \nabla f(x) = \begin{bmatrix}2x & e^x\end{bmatrix} && (4.9)\\
g(x) &= [\quad] && (4.10)\\
b &= [\quad] && (4.11)
\end{aligned}
$$

First we use Problem 3.2 to find the optimal solution x for this problem. We get:

$$\min \quad 0.5 \cdot x^2 + 0.5 \cdot e^x \tag{4.12}$$

The optimal solution for this problem is:

$$0.5 \cdot 2 \cdot x + 0.5 \cdot e^x = 0 \Longrightarrow \widehat{x} = -0.35\dots \tag{4.13}$$

When we combine Equations 4.9 and 4.13 we find:

$$\nabla f(\widehat{x}) = \begin{bmatrix}2 \cdot \widehat{x} & e^{\widehat{x}}\end{bmatrix} = \begin{bmatrix}2 \cdot -0.35 & e^{-0.35}\end{bmatrix} = \begin{bmatrix}-0.70 & 0.70\end{bmatrix} \tag{4.14}$$

Now we forget about the given weights and use inverse multicriteria optimisation to find the optimal weights $\widehat{w}$. The theory gives $w = \widehat{w}$. Problem 4.7 gives:

$$
\begin{array}{ll}
\min_{\epsilon,\gamma} & \phi(\epsilon,\gamma)\\
\text{subject to} & \epsilon = w^T \cdot \nabla f(\widehat{x}) = \begin{bmatrix}-0.70 \cdot w_1 & 0.70 \cdot w_2\end{bmatrix}\\
& w_1 + w_2 = 1\\
& w_i \geq 0 \text{ for } i \in \{1,\dots,n\}
\end{array} \tag{4.15}
$$

So with $w_1 + w_2 = 1 \Longrightarrow w_1 = 1 - w_2$:

$$
\begin{aligned}
\phi(\epsilon,\gamma) = ||\epsilon||_2^2 &= ||\begin{bmatrix}-0.70 \cdot w_1 & 0.70 \cdot w_2\end{bmatrix}||_2^2\\
&= (((-0.70 \cdot w_1)^2 + (0.70 \cdot w_2)^2)^{\frac{1}{2}})^2\\
&= 0.49 \cdot w_1{}^2 + 0.49 \cdot w_2{}^2\\
&= 0.49 \cdot (1 - w_2)^2 + 0.49 \cdot w_2{}^2\\
&= 0.49 \cdot (1 - 2w_2 + w_2{}^2) + 0.49 \cdot w_2{}^2\\
&= 0.49 - 0.98 \cdot w_2 + 0.98 \cdot w_2{}^2
\end{aligned} \tag{4.16}
$$

The optimal solution for this problem is:

$$
\begin{aligned}
-0.98 + 2 \cdot 0.98 \cdot w_2 = 0 &\Longrightarrow 2 \cdot 0.98 \cdot w_2 = 0.98\\
&\Longrightarrow 0.98 \cdot w_2 = 0.5 \cdot 0.98\\
&\Longrightarrow w_2 = 0.5
\end{aligned} \tag{4.17}
$$

So $\widehat{w} = \begin{bmatrix}w_1 & w_2\end{bmatrix} = \begin{bmatrix}1 - w_2 & w_2\end{bmatrix} = \begin{bmatrix}0.5 & 0.5\end{bmatrix}$, which are exactly the weights of the original problem (Problem 4.8).

4.3. IMPLEMENTATION

For the implementation of the methods described in Sections 3.2 and 4.1 we use MATLAB. We started by picking a suitable solver. In our case this is the solver called "quadprog" which can be used for quadratic programming. We have to we rewrite our problem such that we can optimise it with this solver.

This is the kind of problem the solver can optimise:

$$\min_z \tfrac{1}{2} z^T H z + v^T z \text{ such that } \begin{cases} A \cdot z & \leq t \\ Aeq \cdot z & = teq \\ lb & \leq z \leq ub \end{cases}$$

We rewrite Problem 4.7 into a problem that fits the solver. Firstly, in Problem 4.7 we see that w, y, ϵ and γ are all decision variables. We combine these decision variables into the decision variable z:

$$z = \begin{bmatrix} w^T \\ y^T \\ \epsilon \\ \gamma \end{bmatrix} \tag{4.18}$$

The next step is to define the matrix Aeq and the vector teq. Therefore we rewrite the equalities, which are:

$$w^T \nabla f(\hat{x}) + y^T \nabla g(\hat{x}) = \epsilon \tag{4.19}$$

$$y^T (g(\hat{x}) - b) = \gamma \tag{4.20}$$

$$\sum_{i=1}^{n} w_i = 1 \tag{4.21}$$

Equation 4.19 is the same as:

$$w^T \nabla f(\hat{x}) + y^T \nabla g(\hat{x}) - \epsilon = 0.$$

We write this as a matrix multiplication:

$$\begin{bmatrix} \nabla f(\hat{x}) & \nabla g(\hat{x}) & -I & 0 \end{bmatrix} \cdot \begin{bmatrix} w^T \\ y^T \\ \epsilon \\ \gamma \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \tag{4.22}$$

The number of rows of $\begin{bmatrix}\nabla f(\hat{x}) & \nabla g(\hat{x}) & -I & 0\end{bmatrix}$ is the same as the number of decision variables x. The number of columns of $\begin{bmatrix}\nabla f(\hat{x})\end{bmatrix}$ is the same as the number of objective cost-functions. The number of columns of $\begin{bmatrix}\nabla g(\hat{x})\end{bmatrix}$ is the same as the number of objective constraint-functions. I is a diagonal matrix with only ones. The number of columns of I is equal to the number of rows, which is the number of decision variables x. The zero stands for a matrix with only zeros. This matrix has the same number of columns as the number of constraint cost-functions. Concluding the number of rows of $\begin{bmatrix}\nabla f(\hat{x}) & \nabla g(\hat{x}) & -I & 0\end{bmatrix}$ is the sum of the number of objective cost-functions, two times the number of constraint cost-functions and the number of decision variables.

The next equation, Equation 4.20, is the same as:

$$y^T(g(\hat{x})-b)-\gamma=0.$$

We write this as a matrix multiplication:

$$\begin{bmatrix}0 & g(\hat{x})-b & 0 & -I\end{bmatrix}\cdot\begin{bmatrix}w^T\\ y^T\\ \epsilon\\ \gamma\end{bmatrix}=\begin{bmatrix}0\\ \vdots\\ 0\end{bmatrix} \tag{4.23}$$

The number of rows of $\begin{bmatrix}0 & g(\hat{x})-b & 0 & -I\end{bmatrix}$ is the same as the number of constraint cost-functions. The number of columns is the same as in $\begin{bmatrix}\nabla f(\hat{x}) & \nabla g(\hat{x}) & -I & 0\end{bmatrix}$.

The last equality is Equation 4.21. This equation ensures that the sum of the weights is equal to one. In other words, we want to multiply all the weights by one. The sum of these multiplications should equal one. We do this by creating a vector with the same amount of columns as the two vectors above. All the entries are zeros except for the values that will be multiplied by a weight w_i. The number of entries that equal one will be the number of cost-functions. We write this as a matrix multiplication:

$$\begin{bmatrix}1 & \dots & 1 & 0 & \dots & 0\end{bmatrix}\cdot\begin{bmatrix}w^T\\ y^T\\ \epsilon\\ \gamma\end{bmatrix}=\begin{bmatrix}1\end{bmatrix} \tag{4.24}$$

We combine the three matrix multiplications above:

$$\begin{bmatrix}\nabla f(\hat{x}) & \nabla g(\hat{x}) & -I & 0\\ 0 & g(\hat{x})-b & 0 & -I\\ 1\dots1 & 0 & \dots & 0\end{bmatrix}\cdot\begin{bmatrix}w^T\\ y^T\\ \epsilon\\ \gamma\end{bmatrix}=Aeq\cdot z=teq=\begin{bmatrix}0\\ \vdots\\ 0\\ 1\end{bmatrix} \tag{4.25}$$

There are no inequality constraints, but we do have a lower bound for the vector z. Every value of z should be equal or larger than zero.

The goal is to minimise ϵ and γ. More specific, the goal is to minimise $\phi(\epsilon,\gamma) = ||\epsilon||_2^2 + ||\gamma||_2^2$. This is a quadratic equation, without a linear part, so $v^T x$ in the solver should be zero. This results in the following goal function for the solver:

$$\min_z \tfrac{1}{2} z^T Hz + v^T x = \min_z \tfrac{1}{2} z^T Hz + 0 = \min_z \tfrac{1}{2} z^T Hz$$

The last variable we define is H. This is a sparse matrix. In the lower right corner there is an identity matrix. The size of this matrix is the sum of the lengths of the vectors ϵ and γ. The length of the vector ϵ is equal to the number of decision variable x. The length of γ is equal to the number of constraints. This results in the following goal function:

$$\min_z \frac{1}{2} z^T Hz = \min_z \frac{1}{2} \begin{bmatrix} w^T & y^T & \epsilon & \gamma \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{bmatrix} \cdot \begin{bmatrix} w^T \\ y^T \\ \epsilon \\ \gamma \end{bmatrix} \tag{4.26}$$

REFERENCES

[1] S. Breedveld, B. van den Berg, and B. Heijmen, *An interior-point implementation developed and tuned for radiation therapy treatment planning,* Computational Optimization and Applications **68**, 209 (2017).

[2] S. Breedveld, D. Craft, R. van Haveren, and B. Heijmen, *Multi-criteria optimization and decision-making in radiotherapy,* European Journal of Operational Research **277**, 1 (2019).

5
Treatment planning and inverse multicriteria optimisation

Inverse multicriteria optimisation is used for automated treatment planning to determine what the weights of the different OAR or PTV cost-functions should be. This is required to extract useful information based on historical treatment plans.

We use the wish-list given in Table 3.1. Our goal is to find the optimal weights for this problem for an optimal solution $\hat{x}$, while using the wish-list and data matrix A. The patient data that we use to get all the information is from the TROTS data set ([1]).

The linear type cost-functions which compute a pointwise maximum or minimum give goal-functions on each voxel. This results in a different weight for each voxel. Solving such functions with Problem 4.7 is possible but requires additional rewriting to express this problem in canonical form. Therefore, we simplify the case by changing all of the cost-functions of type 1 that compute a pointwise maximum or minimum to a cost-function of type 3 (gEUD), with parameter a equal to thirty. This gives an approximation of the maximum dose. We do not change the linear type cost-functions that compute a mean, because this problem does not occur in those functions. The new wish-list is given in Table 5.1.

5.1. Two objective weights 0.5

To test the method described in the previous chapter, we first try our method with only two objectives, the PTV and one OAR. We repeat the optimisation for all twelve OAR involved. There are no constraints, so the original weights should return (just like the example in Section 4.2). Some of the original weights are really small, that is why we use 0.5 as the weight of the PTV and the OAR. For this problem an optimal solution $\hat{x}$ is determined with the method described in Section 3.2. The optimal weights are determined with inverse multicriteria optimisation. These resulting weights are listed in Table 5.2.

Table 5.1: The modified wish-list. In comparison with the original wish-list in Table 3.1 the cost-functions of type 1 that compute a pointwise maximum or minimum have been changed to a cost-function of type 3 (gEUD), with parameter $a = 30$.

Objectives

Priority	Volume	Type	Goal (Gy)
1	Rectum	3 ($a = 12$)	30
2	Rectum	3 ($a = 8$)	20
3	Rectum	1 (mean)	10
4	External Ring 20 mm	3 ($a = 30$)	31.20
5	PTV Shell 5 mm	3 ($a = 30$)	72.54
6	Anus	1 (mean)	10
7	PTV Shell 15 mm	3 ($a = 30$)	54.60
8	PTV Shell 25 mm	3 ($a = 30$)	39
9	Bladder	1 (mean)	40
10	Hip (L)	1 (mean)	20
10	Hip (R)	1 (mean)	20
11	Unspecified tissue	1 (mean)	100

Constraints

Volume	Type	Limit (Gy)
PTV	3 ($a = 30$)	81.1200
PTV Shell 50 mm	3 ($a = 30$)	39
PTV	4 ($\alpha = 0.8000$ and $d^p = 78$)	0.5000
Rectum	3 ($a = 30$)	78
Hip (L)	3 ($a = 30$)	40
Hip (R)	3 ($a = 30$)	40
Bladder	3 ($a = 30$)	78
Unspecified tissue	3 ($a = 30$)	81.1200
Anus	3 ($a = 30$)	78

In Table 5.2 it is shown that for most of the OAR the results are as we expected: the optimised weights are the same as the original weights. However, for three of the OAR this is different: for the hips and for the external ring. For all three the new weights are 1.000 and 0.000, respectively for the OAR and the PTV. To take a closer look at these results we calculated various values (Table 5.3) and made a dose-volume histogram (DVH) (Figure 5.1) for the weights calculated for the left hip. We do the same for the first rectum cost-function, such that we are able to compare an unexpected result with a result we expected.

To evaluate this we remember the goal of our problem 4.7. Our goal is to minimize the function $\phi(\epsilon,\gamma) = ||\epsilon||_2^2 + ||\gamma||_2^2$. We only use objectives so we want to minimize $||\epsilon||_2^2 = ||w^T \nabla f(x)||_2^2$. For the rectum $||\hat{w}^T \nabla f(\hat{x})||_2^2$ and $||w^T \nabla f(\hat{x})||_2^2$ are the same (Table 5.3). For the left hip the goal-function is higher with the new weights, it doubled.

Table 5.2: The results of inverse multicriteria optimisation for one OAR and one PTV, with starting weights 0.5 and 0.5 ($w = [0.5\ 0.5]$). This is done separately for each OAR. For the rectum objective cost-function the number between the brackets is the goal in Gy for that function.

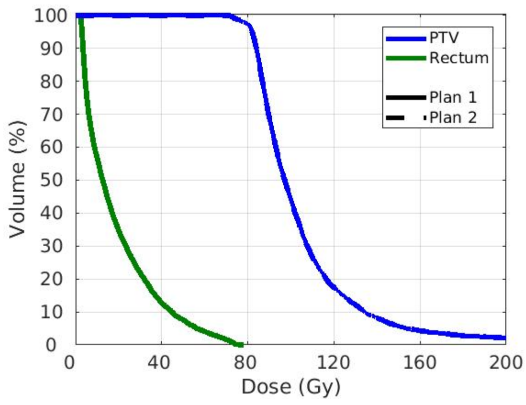
Name OAR	Type	$\widehat{w}_{PTV}$	$\widehat{w}_{OAR}$
Rectum (30)	3	0.5000	0.5000
Rectum (20)	3	0.5000	0.5000
Rectum (10)	1	0.5000	0.5000
External Ring 20 mm	3	0.0000	1.0000
PTV Shell 5 mm	3	0.5000	0.5000
Anus	1	0.5000	0.5000
PTV Shell 15 mm	3	0.5000	0.5000
PTV Shell 25 mm	3	0.5000	0.5000
Bladder	1	0.5000	0.5000
Hip (L)	1	0.0000	1.0000
Hip (R)	1	0.0000	1.0000
Unspecified tissue	1	0.5000	0.5000

Table 5.3: Extra information from the inverse multicriteria optimisation for one OAR and one PTV, with starting weights 0.5 and 0.5 ($w = [0.5\ 0.5]$). There are two OAR in this table: the first rectum cost-function and the cost-function for the left hip. w are the starting weights, $\widehat{w}$ the optimised weights, $\widehat{x}$ the optimal solution for the starting weights w and $x^\star$ the optimal solution for the new weights $\widehat{w}$

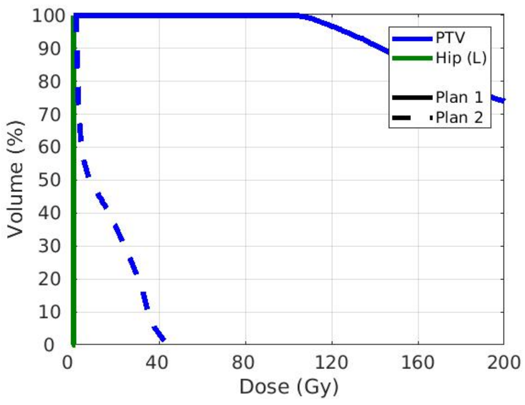
Name OAR	Rectum	Hip (L)
Type	3	1
Objective	30	20
w_{PTV}	0.5000	0.5000
w_{OAR}	0.5000	0.5000
$\widehat{w}_{PTV}$	0.5000	0.000
$\widehat{w}_{OAR}$	0.5000	1.000
$\|\|w^T\nabla f(\widehat{x})\|\|_2^2$	8.523E-5	6.140E-4
$\|\|\widehat{w}^T\nabla f(\widehat{x})\|\|_2^2$	8.523E-5	1.228E-3
$\|\|\widehat{w}^T\nabla f(x^\star)\|\|_2^2$	8.523E-5	1.228E-3

In the DVH in Figure 5.1 we show the old plan and the new plan. The old plan is generated with the old weights w and the corresponding optimal solution $\hat{x}$. For the new plan we need a new optimal solution for x because there are new weights. This solution $x^\star$ is determined with the method described in Section 3.2. After calculating the new optimal solution $x^\star$ the new plan is generated with the new (optimal) weights $\widehat{w}$ and the corresponding optimal solution $x^\star$.

When we take a look at the DVH of the Rectum (Figure 5.1) we see that Plan 1 and Plan 2 are exactly the same. It is difficult to see, but the dotted line (Plan 2) and the continuous line (Plan 1) are on top of each other. This is exactly what we expected.



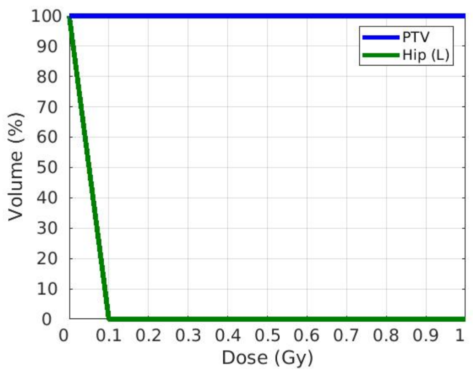
(a) Rectum (30)



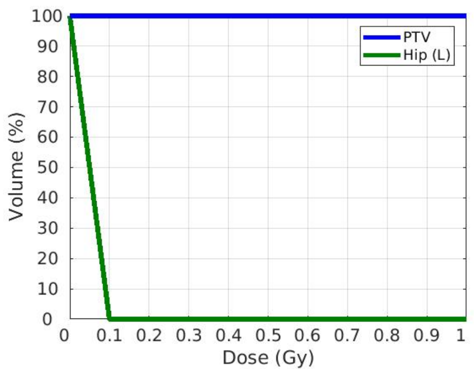
(b) Hip (L)

Figure 5.1: The dose distribution when only using two objectives (PTV and an OAR). For Plan 1 the old weights w are used to get to an optimal solution. For Plan 2 the new weights $\hat{w}$ are used to get to an optimal solution.

When we take a look at the DVH of the left hip (Figure 5.1b) we see that the new plan with the new weights delivers a less dose to the tumor than the first plan with the original weights. This is not what we expected. When zooming in (Figure 5.2) we see that the maximum dose in the left hip is 0.1 Gy (in the original plan (Plan 1) and in the new optimized plan (Plan 2)). In practice this is even smaller, but in a DVH the smallest 'step' possible is of 0.1 Gy. This step is only conforming that there is some dose, but that the dose is negligible. This aplies for Plan 1 and Plan 2. The reason the dose is low is probably because the hips are far from the PTV. A lot of plans will satisfy our optimization, because in almost all plans the dose in the hips is almost equal to zero. That is why it is impossible to get a good plan when only using use one hip and the PTV.



(a) Hip (L) Plan 1



(b) Hip (L) Plan 2

Figure 5.2: A zoomed version of Figure 5.1b

We conclude that our method is working the way we want it to work in most cases. It is not working when the OAR is far from the PTV and therefore not influenced by the radiation of the PTV. In practice there will always be OAR near a PTV. Therefore it does not matter that the optimization is not working when there are no OAR nearby the PTV.

5.2. TWO OBJECTIVES ORIGINAL WEIGHTS

To further inspect the working of our model we repeat the same process, with other weights. We use the original weights of the OAR ([2]). The PTV we use was originally a constraint. We changed it to an objective cost-function. That is why the PTV does not have an original weight. We choose to use 1 as the original weight for the PTV. To make comparing easier we first scale the weights such that the sum of the weights equals one and we calculate the relative error of the weights[1]

Table 5.4: The results of inverse multicriteria optimisation for one OAR and one PTV, with the original weights w and as start weights and the new weights $\hat{w}$.

Name OAR	w_{PTV}	w_{OAR}	$\hat{w}_{PTV}$	$\hat{w}_{OAR}$	rel. error weights
Rectum (30)	1.000	2.634E-12	1.000	5.431E-6	5.431E-6
Rectum (20)	1.000	3.349E-12	1.000	1.429E-5	1.429E-5
Rectum (10)	6.712E-1	3.288E-1	6.712E-1	3.288E-1	1.064E-6
External Ring 20 mm	9.859E-1	1.406E-2	0.000	1.000	9.859E-1
PTV Shell 5 mm	9.760E-1	2.398E-2	9.760E-1	2.398E-2	2.442E-7
Anus	8.540E-1	1.460E-1	8.540E-1	1.460E-1	8.417E-7
PTV Shell 15 mm	9.859E-1	1.413E-2	9.859E-1	1.413E-2	1.472E-7
PTV Shell 25 mm	9.873E-1	1.273E-2	9.873E-1	1.273E-2	1.542E-7
Bladder	1.000	5.682E-12	1.000	1.376E-5	1.376E-5
Hip (L)	9.953E-1	4.658E-3	0.000	1.000	9.953E-1
Hip (R)	9.924E-1	7.625E-3	0.000	1.000	9.924E-1
Unspecified tissue	7.932E-1	2.068E-1	7.932E-1	2.068E-1	1.789E-6

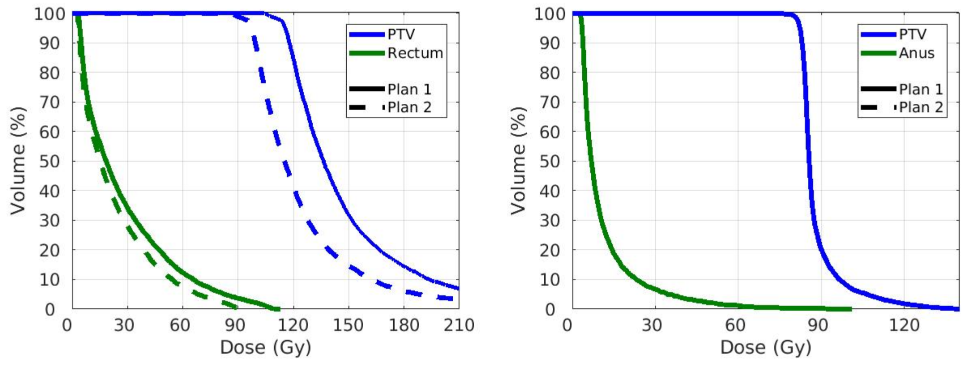
In Table 5.4 we see that we get the same kind of results as in the case where we used 0.5 for the weights. In Table 5.5 we see that although the change of the weights is small the goal function ($\phi(\epsilon,\gamma)$) can change a lot. For example, for the first rectum OAR the relative error of the weights is only $5.431 \cdot 10^{-6}$, but the goal function is more than $6 \cdot 10^{7}$ times as large. This indicates that although we can optimise the weights pretty well, there is too much noise, due to the finite precision of real numbers used in computing, to arrive at the exact answer we are looking for.

[1] relative error of the weights = $\sum_{i=1}^{n} \frac{w_i - \hat{w}_i}{n}$ (with n the number of objective cost-functions)

Table 5.5: The relative error of the weights of inverse multicriteria optimisation for one OAR and one PTV, and the goal function $\phi(\epsilon,\gamma) = ||w^T \nabla f(x)||_2^2$. With the original weights w and optimal solution $\hat{x}$, with the new weights $\hat{w}$ and the old optimal solution $\hat{x}$ and with the new weights $\hat{w}$ and the new optimal solution $x^\star$

Name OAR	rel. error weights	$\|\|w^T \nabla f(\hat{x})\|\|_2^2$	$\|\|\hat{w}^T \nabla f(\hat{x})\|\|_2^2$	$\|\|\hat{w}^T \nabla f(x^\star)\|\|_2^2$
Rectum (30)	5.431E-6	4.522E-16	2.709E-8	9.284E-10
Rectum (20)	1.429E-5	8.137E-16	6.213E-8	3.466E-9
Rectum (10)	1.064E-6	6.599E-4	6.599E-4	6.599E-4
External Ring 20 mm	9.859E-1	3.775E-5	2.685E-3	3.429E-3
PTV Shell 5 mm	2.442E-7	1.964E-05	1.964E-5	1.964E-5
Anus	8.417E-7	4.505E-4	4.505E-4	4.505E-4
PTV Shell 15 mm	1.472E-7	9.335E-6	9.335E-6	9.335E-6
PTV Shell 25 mm	1.542E-7	5.841E-6	5.841E-6	5.841E-6
Bladder	1.376E-5	7.503E-15	3.644E-8	1.822E-8
Hip (L)	9.953E-1	5.721E-6	1.228E-3	1.228E-3
Hip (R)	9.924E-1	9.464E-6	1.241E-3	1.241E-3
Unspecified tissue	1.789E-6	3.201E-5	3.201E-5	3.201E-5

To take a look at the dose distribution we calculate the new optimal solution $x^\star$ for the new weights $\hat{w}$. Plan 1 is the optimal dose distribution for the old weights w and Plan 2 is the optimal dose distribution for the new weights $\hat{w}$. We compare two different situations in Figure 5.3. Firstly, we want to see the influence of the small weight change for the first rectum cost-function. That is why we make a DVH for the optimisation with the PTV cost-function and the first rectum cost-function. Secondly, we want to compare this unexpected result with a result we did expect. We expected the goal functions to be equal, such as for the optimisation with the Anus and the PTV cost-functions. This is the second situation we look at.



(a) Rectum (30) (b) Anus

Figure 5.3: The dose distribution when only using two objectives. For Plan 1 the old weights w are used to get to an optimal solution, Plan 2 is the plan the new weights $\hat{w}$ are used to get to an optimal solution.

Despite only a small difference between the old weights and the new weights for the rectum (30) and PTV cost-function, the dose distribution for the plans containing the rectum (30) cost-function and the PTV cost-function is visibly different. With the information from Section 2.2 we take a better look at Figure 5.3. We see a small improvement for the OAR (the rectum (30)). A smaller part of the rectum gets a higher dose. But there is also a downside to the new plan: a smaller part of the PTV gets a high dose. There is a larger difference for the PTV and that difference is negative so Plan 1 (the original plan) seems to be better than Plan 2 (the new plan).

For the Anus there is no difference between the two plans. This is exactly what we expected, because the goal function is the same for the old and new weights. When we look at the relative error of the weights the error is only 6.5 times larger for the rectum (30). This is quite a small difference but it influences the solution a lot, looking at Figure 5.3. Again this indicates that although we can optimise the weights pretty good, there is too much noise to arrive at the exact treatment plan that we are looking for.

5.3. MULTIPLE OBJECTIVES

In reality there is more than one OAR. We are going to add one OAR at a time. We start with only one OAR (the first rectum objective function) and the PTV. Then, we add the second OAR (the second rectum objective function), and so on. The weights we use are the original weights of the problem and for the PTV we set the weight to one.

Table 5.6: The relative error of the weights of inverse multicriteria optimisation for the OAR above, the newly added OAR and the PTV. With the original weights w as starting weights and the new weights $\hat{w}$. The goal function is $\phi(\epsilon,\gamma) = ||\epsilon||_2^2 = ||w^T\nabla f(x)||_2^2$, with the original weights w and optimal solution $\hat{x}$, with the new weights $\hat{w}$ and the old optimal solution $\hat{x}$ and with the new weights $\hat{w}$ and the new optimal solution $x^\star$.

Added OAR	# OAR	rel. error weights	$\|\|w^T\nabla f(\hat{x})\|\|_2^2$	$\|\|\hat{w}^T\nabla f(\hat{x})\|\|_2^2$	$\|\|\hat{w}^T\nabla f(x^\star)\|\|_2^2$
Rectum (30)	1	5.431E-6	4.521E-16	2.709E-8	9.284E-10
Rectum (20)	2	4.826E-6	1.277E-15	3.372E-8	1.517E-9
Rectum (10)	3	8.044E-3	6.599E-4	6.567E-4	6.245E-4
Ext. Ring 20 mm	4	2.657E-1	1.838E-2	2.116E-3	1.585E+7
PTV Shell 5 mm	5	7.059E-2	6.080E-4	4.541E-4	5.183E-4
Anus	6	8.098E-2	7.259E-4	5.281E-4	9.198E-4
PTV Shell 15 mm	7	6.762E-2	7.122E-4	5.235E-4	8.851E-4
PTV Shell 25 mm	8	4.962E-2	6.997E-4	5.133E-4	8.708E-4
Bladder	9	4.889E-2	6.997E-4	4.855E-4	8.212E-4
Hip (L)	10	5.850E-2	6.963E-4	4.806E-4	1.141E-3
Hip (R)	11	5.484E-2	6.912E-4	4.798E-4	4.849E-4
Unspecified tissue	12	6.252E-2	6.036E-4	3.589E-4	1.820E-4

The new weights $\widehat{w}$ and the old weights w can be found in Appendix A.

In Table 5.6 we see that for the first two problems $||w^T \nabla f(\hat{x})||_2^2$ is less than $||\hat{w}^T \nabla f(\hat{x})||_2^2$. This means that for our problems with one or two objective cost-functions and the PTV the old weights w give a better outcome. By a better outcome we mean that we are closer to the goal of our optimisation problem. In Section 4.1 we stated that our goal is to minimise $\phi(\epsilon,\gamma) = ||\epsilon||_2^2 = ||w^T \nabla f(x)||_2^2$.

For the other problems we see something different in Table 5.6. $||w^T \nabla f(\hat{x})||_2^2$ is larger than $||\hat{w}^T \nabla f(\hat{x})||_2^2$. This means that for the problems with more than three objective cost-functions and the PTV the new weights $\widehat{w}$ give a better outcome.

For the problems with four, five, six, seven, eight or nine OAR there is something else that stands out. After finding the new weights $\hat{w}$ we try to find the new optimal solution $x^\star$ for the new weights $\hat{w}$. This new solution $x^\star$ gives a worse result for our goal function than the optimal solution $\widehat{x}$ for our old weights w. Again, with a better outcome we mean that our goal function is minimised as well as possible. The problem with four OAR and the PTV gives the worst result: $|||\hat{w}^T \nabla f(x^\star)||_2^2$ is much larger than all of the other results.

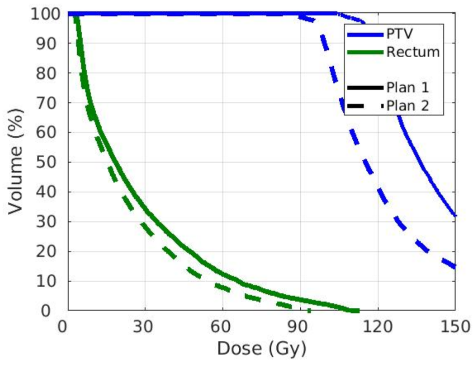
When we take a closer look at the optimisation process we see that the solver has difficulties finding an optimal solution for these problems. After two hundred iterations (which is the maximum number of iterations we allowed) the optimisation stops and delivers an 'optimal' solution $x^\star$. This is the best solution the solver could find in the number of iterations we allowed, but it does not give a solution that meets our demands.

To take a closer look at our unexpected results we look at the DVHs for the problems in the Figures 5.4 and 5.5. Firstly, it stands out that the DVH for the PTV with all three of the objective cost-functions for the rectum gives exactly the same plan. This is not what we expected because our goal function is significantly less well minimised.

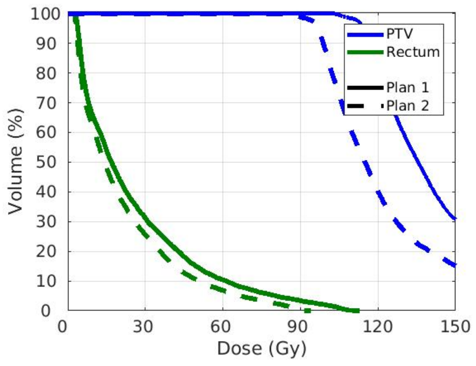
Secondly, we see a trend. For different OAR a bigger part of the OAR gets a relatively small dose, but the percentage of OAR that gets a relatively high dose is larger. It seems like we minimise more on the maximum dose than on the average dose, but this was not our goal.

Thirdly, it stands out that when all the OAR are added, in the last situation with twelve OAR, we see that the dose distribution for the PTV is almost the same in Plan 1 and Plan 2. However, the dose distribution for the OAR differs.

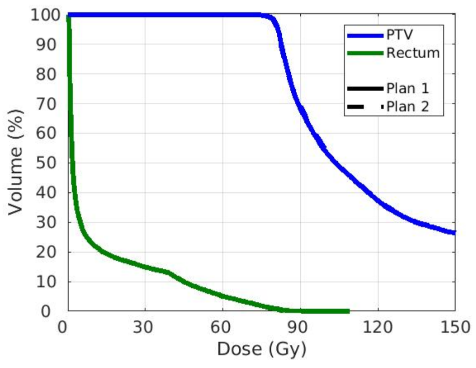
We conclude that using inverse multicriteria optimisation for treatment planning gives a good result for the dose distribution for the PTV. However, the dose distribution for the OAR gets worse. This is not what we expected and hoped for as an outcome.



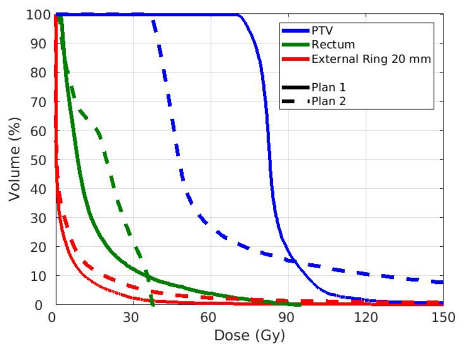
(a) PTV and one OAR



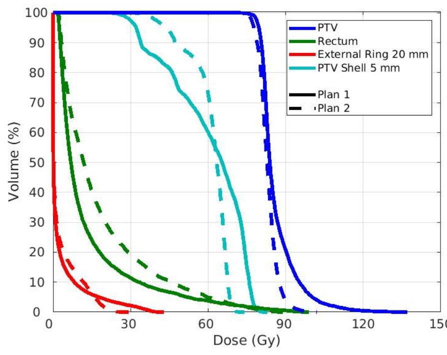
(b) PTV and two OAR



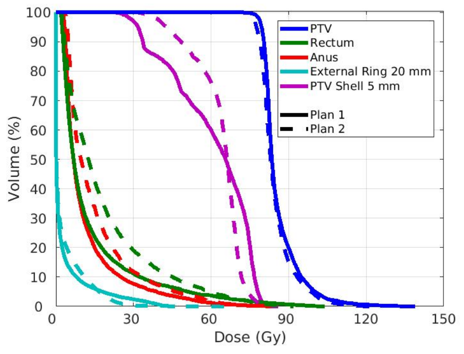
(c) PTV and three OAR



(d) PTV and four OAR

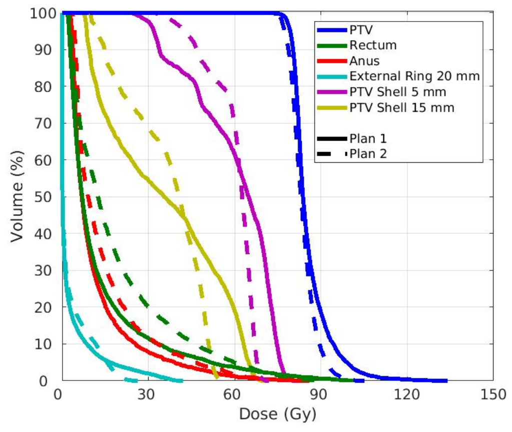


(e) PTV and five OAR

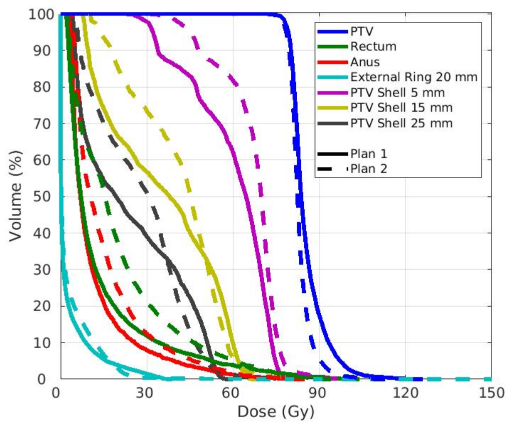


(f) PTV and six OAR

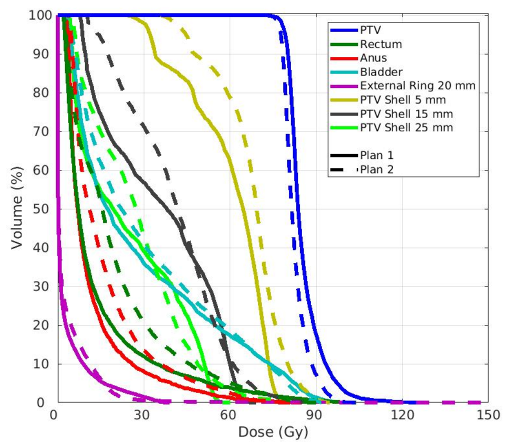
Figure 5.4: The dose distribution when using the PTV and one or more OAR. Just like in Table 5.6. For Plan 1 the old weights w are used to get to an optimal solution, Plan 2 is the plan the new weights $\hat{w}$ are used to get to an optimal solution. Part 1.



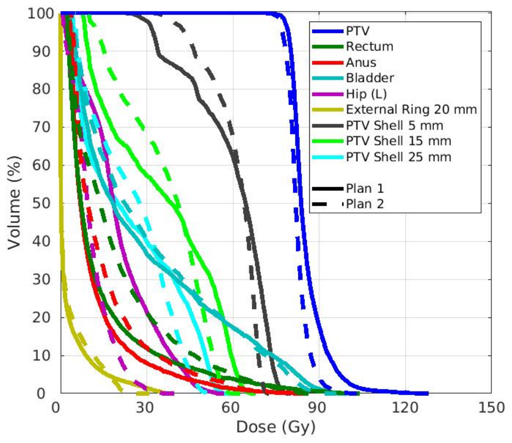
(a) PTV and seven OAR



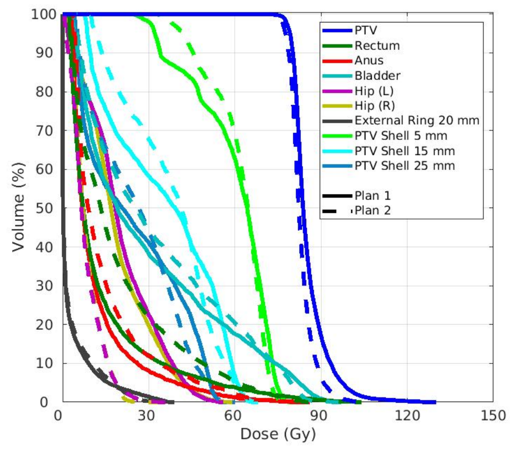
(b) PTV and eight OAR



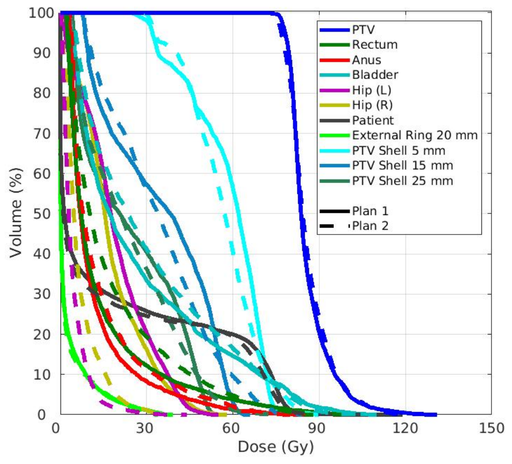
(c) PTV and nine OAR



(d) PTV and ten OAR



(e) PTV and eleven OAR



(f) PTV and twelve OAR

Figure 5.5: The dose distribution when using the PTV and one or more OAR. Just like in Table 5.6. For Plan 1 the old weights w are used to get to an optimal solution, Plan 2 is the plan the new weights $\hat{w}$ are used to get to an optimal solution. Part 2.

REFERENCES

[1] S. Breedveld and B. Heijmen, *Data for trots – the radiotherapy optimisation test set,* Data in Brief **12**, 143 (2017).

[2] S. Breedveld, P. R. M. Storchi, and B. J. M. Heijmen, *The equivalence of multi-criteria methods for radiotherapy plan optimization,* Physics in Medicine and Biology **54**, 7199 (2009).

6 LINEAR INVERSE MULTICRITERIA OPTIMISATION

The results for the non-linear problems behave unexpectedly, and the errors are larger than what we hoped for. That is why we take a look at a more simple problem, a linear problem.

6.1. WISHLIST

The linear problem we are going to analyse is a wish-list (Table 6.1) for a tumour (clinical target volume (CTV)) in the head-and-neck area. For this study, we use a simplified wish-list, where only the minimum and maximum dose to the tumour are constrained.

Table 6.1: Wish-list linear problem

Objectives

Priority	Volume	Type
3	Parotid (L)	1 (min mean)
3	Parotid (R)	1 (min mean)
4	SMG (L)	1 (min mean)
4	SMG (R)	1 (min mean)
6	SCM	1 (min mean)
6	MCM	1 (min mean)
6	MCI	1 (min mean)
6	MCP	1 (min mean)
7	Larynx	1 (min mean)
8	Oral Cavity	1 (min mean)

Constraints

Volume	Type	Limit (Gy)
CTV	1 (min)	64.68
CTV	1 (max)	69.96

Some of the OAR are well known such as the larynx and the oral cavity. The other OAR need some explanation.

- Parotids are the salivary glands in front of the ears, also known as the large salivary glands.
- Submandibular glands (SMGs) are the salivary glands located in the bottom of the mouth, also known as the small salivary glands.
- Four muscles that are used to swallow: musculus constrictor superior (MCS), musculus constrictor medius (MCM), musculus constrictor inferior (MCI) and the musculus constrictor cricopharyngeus (MCP).

6.2. OPTIMISATION PROBLEM

Our goal is still the same: we want to find the optimal weights for the problem ([1]). Again we will look at an convex problem. Therefore, we expect the starting weights to be equal to the weights we find after optimising the problem. We will use multicriteria optimisation and inverse multicriteria optimisation, but this time we do not use the Lagrangian. We start with our linear problem (Problem 6.1). In section 6.3 we will explain how the optimal solution $\widehat{x}$ can be determined.

$$\begin{array}{ll} \min_x & w^T Cx \\ \text{subject to} & Ax \geq b \\ & x \geq 0 \end{array} \tag{6.1}$$

There are multiple variables in this problem. Firstly, C ($m \times n$ matrix) contains the objective functions. Secondly, $w \in \mathbb{R}^m$ this vector contains the weights for all of the objective functions, similar to the weights w_i in Problem 3.2. Thus, m is the number of OAR. Thirdly, $x \in \mathbb{R}^n$ our decision variables, similar to the decision variables x in Problem 3.2. This vector contains all information about the a specific beamlet of the treatment device, for example the intensity and angle of the beamlet. Thus, n is the number of decision variables. Fourthly A (a $k \times n$ matrix) is the data matrix. This data matrix contains the data about the tumour, which is different from the data matrix described in Section 3.3. Finally $b \in \mathbb{R}^k$, contains the lower and upper bound for the dose delivered in the tumour.

In short the problem can be divided into three parts. Firstly, $w^T Cx$, where Cx is a $m \times 1$ vector which contains the mean dose for each OAR. Therefore, when minimising $w^T Cx$ we minimise the weighted sum of the mean dose in every OAR. Secondly, $Ax \geq b$ gives a lower and upper bound for the dose delivered in the tumour. Thirdly, $x \geq 0$ gives that the decision variables should all be non-negative.

To find our optimal weights $\widehat{w}$ for our optimal solution $\widehat{x}$ we need to formulate an inverse optimisation problem. Firstly, we determine the dual of Problem 6.1: this is Problem 6.2 (with y the duality vector, corresponding with $Ax \leq b$).

$$\begin{array}{ll} \max_y & b^T y \\ \text{subject to} & C^T w \geq A^T y \\ & y \geq 0 \end{array} \tag{6.2}$$

We get our inverse optimisation problem (Problem 6.3) when we write Problem 6.2 in the form of an absolute duality gap minimization problem.

$$\begin{array}{ll} \min_{y,w} & w^T C\hat{x} - b^T y \\ \text{subject to} & C^T w \geq A^T y \\ & y \geq 0 \\ & w \geq 0 \end{array} \tag{6.3}$$

Now the optimal weights $\hat{w}$ can be determined, this will be explained in Section 6.3.

6.3. IMPLEMENTATION

For the implementation of the methods described in Section 6.2 we use MATLAB. This time we did not use the "quadprog" solver. Because of the linearity of the problems in Section 6.2 we used a solver called "linprog". This solver is suited for linear programming. We have to rewrite our problem such that we can optimise it with this solver.

This the kind of problem the solver can optimise:

$$\min_z f^T z \text{ such that } \begin{cases} V \cdot z & \leq t \\ Veq \cdot z & = teq \\ lb & \leq z \leq ub \end{cases}$$

We start with the implementation of Problem 6.1. The vector of decision variables x will be the z in the implementation. f is equal to $(w^T C)^T$, because this gives the goal function we want:

$$\min_z f^T z = ((w^T C)^T)^T x = w^T C x$$

In Problem 6.1 we have: $Ax \geq b$. Multiplying this with -1 gives: $-Ax \leq -b$. This is the formulation that is asked for by the solver and this results in: $V = -A$ and $t = -b$. We do not have an "equal to" constraint, so Veq and teq will be empty vectors. There is no upper bound for our decision variables x, but there is a lower bound, namely zero. This gives us everything to get our optimal solution $\hat{x}$ using the "linprog" solver.

For the implementation of Problem 6.3 we also have to make some modifications, such that our problem is in the format of the solver. Again, we start with defining the decision variable z:

$$z = \begin{bmatrix} w \\ y \\ \epsilon \\ \gamma \end{bmatrix} \tag{6.4}$$

Secondly we define Veq and teq these will form the equality conditions. We want ϵ to equal $C\hat{x}$ and γ equal to $-b^T y$. This is achieved by using the equations $C\hat{x}-\epsilon = 0$ and $-b^T y - \gamma = 0$. We add one more equality to avoid the trivial solution $sum(w) = 0$, we set the sum of w equal to one.

$$\begin{bmatrix} C\hat{x}^T & 0\dots0 & -1 & 0 \\ 0\dots0 & b^T & 0 & -1 \\ 1\dots1 & 0\dots0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} w \\ y \\ \epsilon \\ \gamma \end{bmatrix} = Veq \cdot z = teq = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \tag{6.5}$$

Thirdly, we define V and t these will form the inequality conditions. Problem 6.3 gives $C^T w \geq A^T y$. To make the inequality fit the solver we subtract $A^T y$ on both sides of the inequality and we multiplying the inequality with minus one: $-C^T w + A^T y \leq 0$.

$$\begin{bmatrix} -C^T & A^T & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} w \\ y \\ \epsilon \\ \gamma \end{bmatrix} = V \cdot z \leq t = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \tag{6.6}$$

6

REFERENCES

[1] A. Babier, J. J. Boutilier, M. B. Sharpe, A. L. McNiven, and T. C. Y. Chan, *Inverse optimization of objective function weights for treatment planning using clinical dose-volume histograms,* Physics in Medicine & Biology **63**, 105004 (2018).

7 Treatment planning and linear inverse multicriteria optimisation

Similar as in Chapter 5 we use inverse multicriteria optimisation to determine the optimal weights $\hat{w}$ for the different OAR and PTV cost-functions for an optimal solution $\hat{x}$. The difference is that we will look at a completely linear problem. Therefore, we will use the wish-list given in Table 6.1. All the cost-functions that are used to optimise a treatment plan for this wish-list are linear functions that compute a mean dose.

7.1. Equal weights

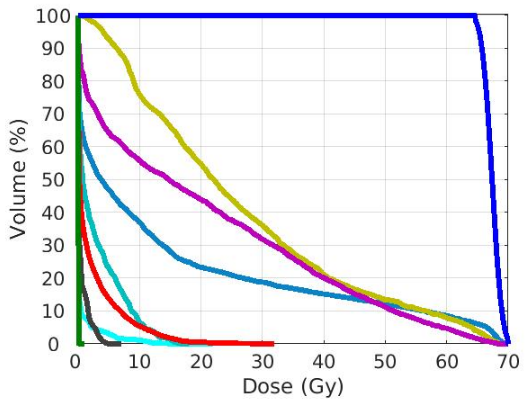
To test the method described in the previous chapter, we start with equal weights for every cost-function. We use these weights, our wish-list (Table 6.1) (for this patient the swallowing muscle MCP is not on the wish-list) and the implementation of our problem (Section 6.3) to get an optimal solution $\hat{x}$ for our problem. After that we optimise the weights to get the optimal weights $\hat{w}$ for our optimal solution $\hat{x}$.

We analyse there sults by looking at the goal function, $w^T C\hat{x} - b^T y$, and the DVH. To generate a DVH we need a new optimal solution $x^\star$. We get that solution by finding the optimal solution for the problem with the new weights $\hat{w}$.

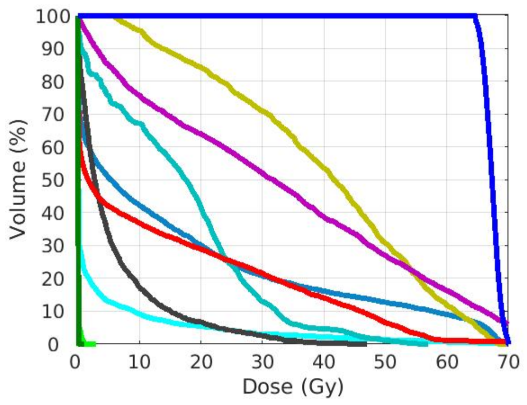
Table 7.1: The goal function of our problem described in Section 7.1. With $\hat{x}$ the optimal solution for the problem with the old weights w and $x^\star$ the optimal solution for the problem with the new weights $\hat{w}$.

Goal function	Result
$w^T C\hat{x} - b^T y$	2.851E-13
$\hat{w}^T C\hat{x} - b^T \hat{y}$	0
$\hat{w}^T C x^\star - b^T \hat{y}$	0

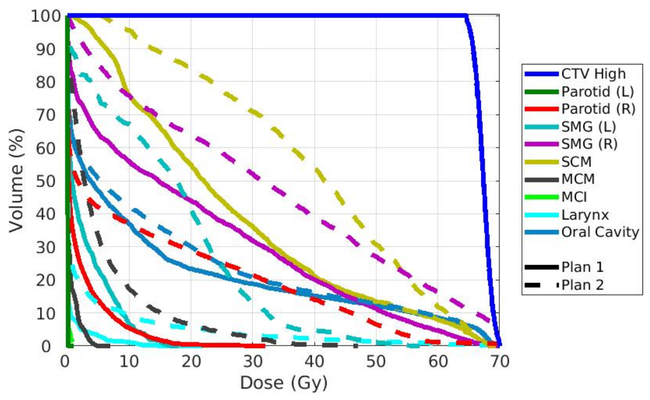
In Table 7.1 it stands out that the goal functions with the new weights $\hat{w}$ are exactly equal to zero. Normally this would not happen when we optimise a problem, because zero is too specific. We get really close but not to zero exactly.



(a) Plan 1



(b) Plan 2



(c) A comparison of the two plans.

Figure 7.1: The dose distribution for the problem described in Section 7.1. For Plan 1 the old weights w are used to get to an optimal solution $\hat{x}$. For Plan 2 the new weights $\hat{w}$ are used to get to an optimal solution $x^\star$.

In Figure 7.1 we see that Plan 1 (the old weights w are used to get to an optimal solution $\hat{x}$) differs a lot from Plan 2 (the new weights $\hat{w}$ are used to get to an optimal solution $x^\star$). To investigate these results and the unexpected results of the goal-functions we take a look at $C\hat{x}$, $Cx^\star$, the old weights w and the new weights w (Table 7.2).

Table 7.2: $C\hat{x}$, $Cx^{\star}$, the old weights w and the new weights w for the problem described in Section 7.1. With $\hat{x}$ the optimal solution for the problem with the old weights w and $x^{\star}$ the optimal solution for the problem with the new weights $\hat{w}$.

OAR	$C\hat{x}$	$Cx^{\star}$	w	$\hat{w}$
Parotid (L)	0.000	0.000	0.111	1
Parotid (R)	1.972	1.381E+1	0.111	0
SMG (L)	2.793	1.682E+1	0.111	0
SMG(R)	2.053E+1	3.335E+1	0.111	0
SCM	2.613E+1	3.967E+1	0.111	0
MCM	4.430E-1	5.641	0.111	0
MCI	0.000	3.857E-2	0.111	0
Larynx	4.122E-1	3.192	0.111	0
Oral Cavity	1.447E+1	1.624E+1	0.111	0

The reason for the differences between the plans comes to light. Some of the values of $C\hat{x}$ (and also $Cx^{\star}$) are zero. This implies that some OAR are so far removed from the tumour that they do not receive any dose at all. If we try to find the optimal weights $\hat{w}$, the total weight shifts to one of the cost-functions equal to zero. This gives a good result because our goal function will be equal to zero. This does not give us the kind of plan we are looking for, because when we try to find a new optimal solution $x^{\star}$ all other cost-functions are not included in the optimisation because their weights are equal to zero. That is why we conclude that the OAR for which $C\hat{x}$ is equal to zero are irrelevant for our optimisation.

7.2. EQUAL WEIGHTS WITHOUT IRRELEVANT OAR

To solve this problem we calculate $C\hat{x}$. If $C\hat{x}$ is zero for a certain OAR, we do not take this OAR into account when we calculate the optimal weights $\hat{w}$. After calculating the optimal weights $\hat{w}$ we add zeros as the weights for the OAR cost-functions for which $C\hat{x}$ was equal to zero. It does not matter what the weights are that we assign to these cost-functions, because they won't receive any dose.

After the adjustments we optimise the problem for nineteen different patients. For the nineteen patients we take a look at a part of the data obtained:

- The relative error between the decision variables $\hat{x}$ and $x^{\star}$.
- The relative error between the weights w and $\hat{w}$.
- The goal function $w^T C\hat{x} - b^T y$. For three different combinations: the start weights w and it's optimal solution $\hat{x}$, the new weights w and the optimal solution $\hat{x}$ and the new weights w and it's optimal solution $x^{\star}$.
- The minimum value of $C\hat{x}$ (the vector which contains the mean dose for each OAR).

The data is shown in Table 7.3.

Table 7.3: Data obtained of 19 optimisation problems which are as described above. We used equal weights for the OAR as starting weights.

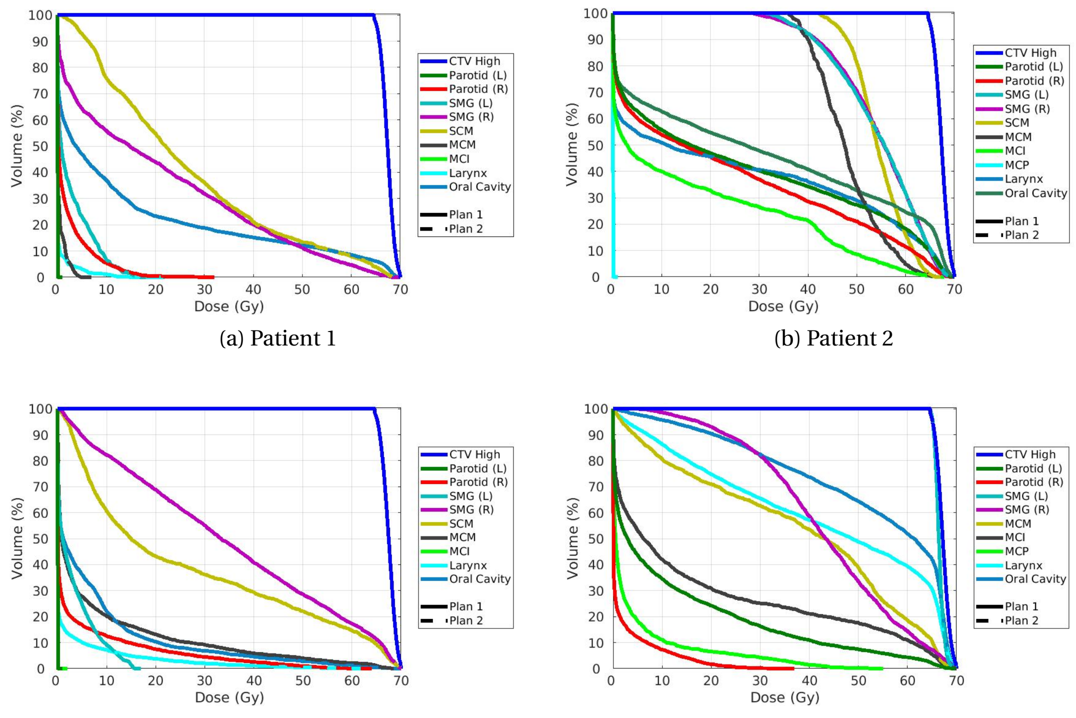
patient	rel. error x	rel. error w	$w^T C\hat{x} - b^T y$	$\hat{w}^T C\hat{x} - b^T \hat{y}$	$\hat{w}^T Cx^\star - b^T \hat{y}$	min$(C\hat{x})$
1	3.585E-1	9.856E-3	2.119	7.105E-14	-5.684E-14	0.000
2	9.161E+1	2.000E-2	-1.266E-9	-1.436E-9	-2.132E-12	1.973E-22
3	1.735E+1	1.011E-4	-1.926E-12	1.375E-12	-2.629E-13	1.299E-2
4	9.687E-2	3.333E-2	1.481	8.260E-14	9.948E-14	0.000
5	1.143E+2	2.222E-2	1.425	-3.759E-12	-2.835E-12	0.000
6	2.591E+1	7.929E-4	2.533	-2.270E-12	-3.369E-10	0.000
7	8.252E+1	8.497E-4	1.900	9.557E-13	-7.674E-13	0.000
8	1.348	1.009E-1	1.167	5.240E-14	1.776E-14	0.000
9	9.595E+1	2.379E-4	3.553E-13	1.279E-13	-5.255E-10	6.577E-1
10	4.260E+1	3.217E-4	2.445	-5.455E-7	-5.590E-7	0.000
11	2.501E+1	1.688E-3	1.220	-5.915E-13	-9.432E-13	0.000
12	4.556	1.068E-3	2.650	-3.880E-9	-1.341E-8	0.000
13	1.516E+2	2.255E-4	-2.963E-9	-2.978E-9	7.937E-12	1.897
14	1.234E-1	8.439E-4	2.602	-1.456E-9	-3.700E-10	0.000
15	5.956E-1	1.272E-3	1.338	-1.300E-12	-6.839E-13	0.000
16	5.156E-1	7.844E-3	1.351	1.318E-12	7.141E-13	0.000
17	1.787	2.803E-3	1.075E-13	-3.610E-12	-1.118E-10	2.113E-2
18	4.852E+1	2.778E-2	1.247	-2.416E-13	-5.347E-13	0.000
19	2.356E+1	1.152E-2	1.675	7.105E-14	-7.816E-14	0.000

For most of the problems our goal functions are minimised and although some values of $C\hat{x}$ are very low the optimisation seems to have succeeded for almost all of our problems. There are a few problems for which the data stands out:

- Problem 2 has a really small value in the vector Cx. Maybe this will result in what we saw before: the total weight shifts to this OAR.
- For problem 8 we get a larger relative error for w than for the other problems.
- For problem 13 we get a larger relative error for x than for the other problems.

We take a closer look at the DVHs of the problems mentioned above. We compare these DVHs with the DVH of a problem for which the data looks like it will give a similar DVH for $\hat{x}$ and $x^\star$. This means that we see nothing outstanding in the data for this problem. We choose problem 1. These DVHs are shown in Figure 7.2.

When we take a look at Figures 7.2 we see that Plan 1 and Plan 2 are exactly the same. It is difficult to see, but the dotted line (Plan 2) and the continuous line (Plan 1) are on top of each other.



(a) Patient 1 (b) Patient 2

(c) Patient 8 (d) Patient 13

Figure 7.2: The dose distribution for different patients from Table 7.3. For Plan 1 the old weights w are used to get to an optimal solution $\hat{x}$. For Plan 2 the new weights $\hat{w}$ are used to get to an optimal solution $x^{\star}$.

From the DVHs we conclude that although sometimes the new weights solution differ a lot from the old weights (the relative error is not always small), the combination of the old weights with its optimal solution gives the same dose distribution as the new weights with its optimal solution. This can be explained because the new optimal solution $x^{\star}$ and the old optimal solution $\hat{x}$ also differ a lot sometimes.

We remember that every x contains the information about a specific beamlet of the Cyberknife, such as the intensity and angle of the beamlet. For example, if the change in angle or intensity of one beamlet results in 10 Gy less dose in 50 percent of the OAR and the change in another beamlet results in 10 Gy more dose in 50 precent of the OAR there is not an difference in the dose distribution. The same percentage of an OAR receives the same amount of dose. The only difference is the beamlet that irradiates the dose. This does not make a difference for the quality of our generated treatment plan.

It also stands out that the small value in the vector $C\hat{x}$ of problem 2 did not cause any differences between Plan 1 and Plan 2 .

We conclude that for all of our problems inverse multicriteria optimisation was successful. We learned that only if our goal-function is zero, something went wrong. Thus, that can be a test to see if the optimisation succeeded. This will take less time then generating all of the DVHs.

7.3. ORIGINAL WEIGHTS

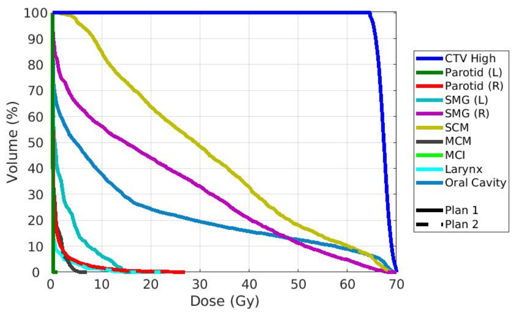
To test if it also works for more realistic weights we change our starting weights. We start with our original weights. The reason this could fail is that those weights can be really small or large. This could lead to the same kind of situations as the small value of $C\hat{x}$ for an OAR.

From Section 7.1 we learned that we only have to check the results of our goal-functions. In Table 7.4 we see that none of the goal-functions is equal to zero. So there is no problem where one OAR got the total weight. To test this we look at the DVHs for all of the patients in Figures 7.3, 7.4 and 7.5. We see that Plan 1, for which the old weights w are used to get to an optimal solution, and Plan 2, for which the new weights $\hat{w}$ are used to get to an optimal solution, are exactly the same.

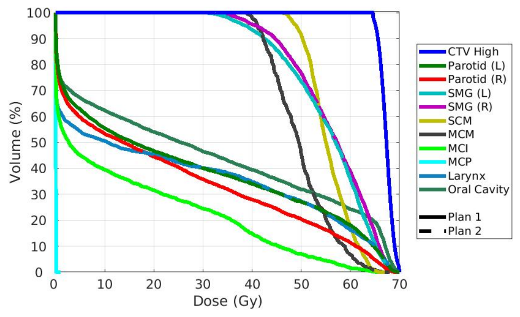
We conclude that our conclusion at the end of Section 7.2 is still correct. For all of our linear problems inverse multicriteria optimisation was successful. Only if our goal-function is zero something went wrong. Thus, that can be a test to see if the optimisation succeeded. This will take less time then generating all of the DVHs.

Table 7.4: Obtained data of 19 optimisation problems which are as described above. We used the original weights of the OAR as start weights.

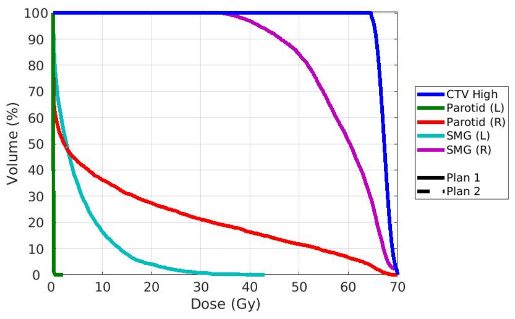
patient	$w^T C\hat{x} - b^T y$	$\hat{w}^T C\hat{x} - b^T \hat{y}$	$\hat{w}^T Cx^\star - b^T \hat{y}$
1	1.793	1.955E-12	3.997E-15
2	1.369E+1	-7.251E-11	-1.343E-9
3	3.635	-6.138E-7	-6.378E-7
4	1.054	1.660E-14	-2.637E-15
5	7.970	-2.593E-13	-2.949E-13
6	9.526	-7.217E-8	-8.552E-8
7	4.855	-5.457E-7	-7.257E-7
8	2.392	9.059E-14	4.574E-14
9	5.436	-3.582E-11	-1.995E-9
10	1.093E+1	-1.172E-12	-2.736E-12
11	3.799	2.327E-13	1.092E-13
12	1.178E+1	-1.830E-13	-2.238E-13
13	1.327E+1	-7.645E-12	1.402E-11
14	2.653	6.573E-14	-1.861E-10
15	3.704	1.242E-12	-3.961E-13
16	3.373	1.759E-13	-4.068E-13
17	3.874	2.345E-13	-1.972E-13
18	9.109E-1	-1.927E-13	-5.596E-14
19	3.622	1.399E-14	-9.104E-14



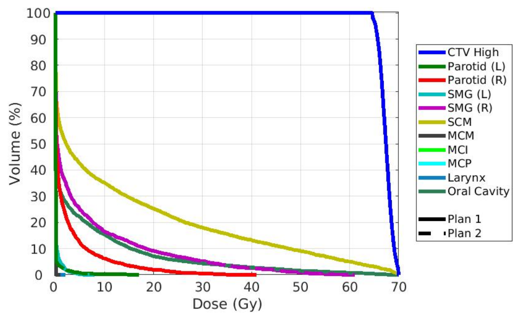
(a) Patient 1



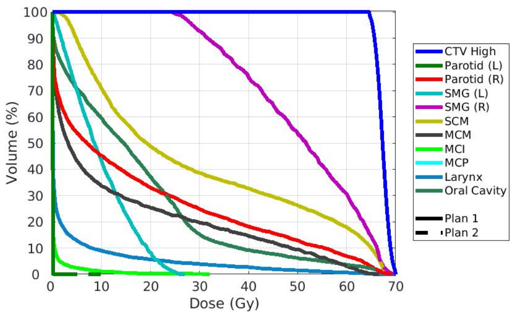
(b) Patient 2



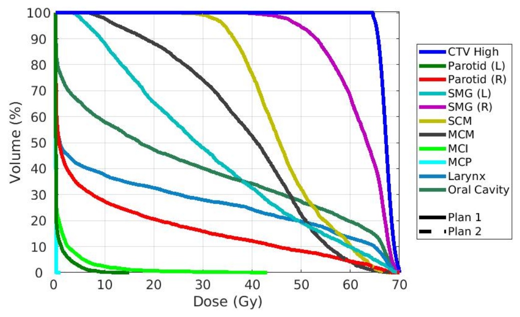
(c) Patient 3



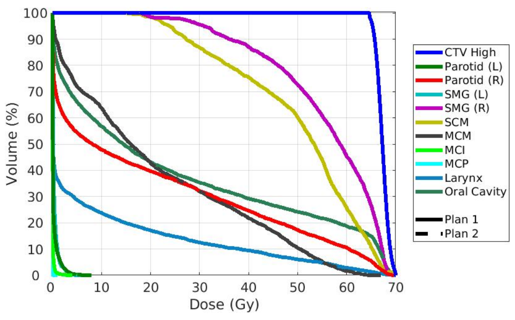
(d) Patient 4



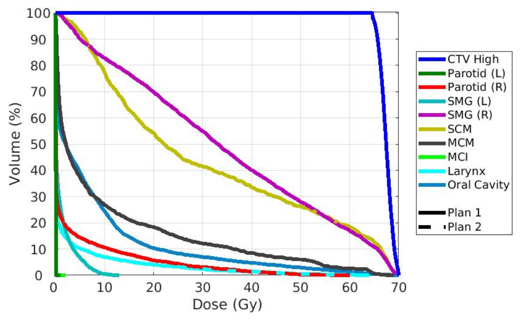
(e) Patient 5



(f) Patient 6

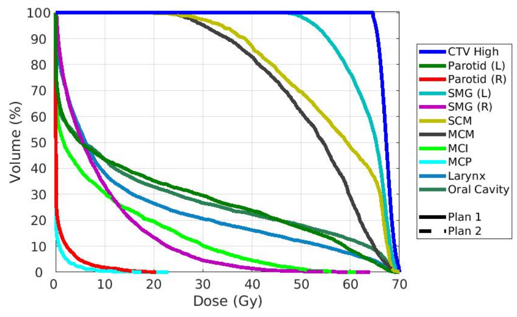


(g) Patient 7

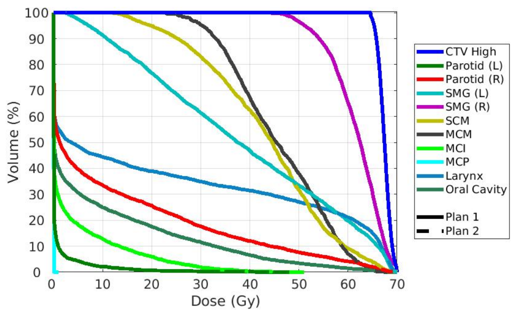


(h) Patient 8

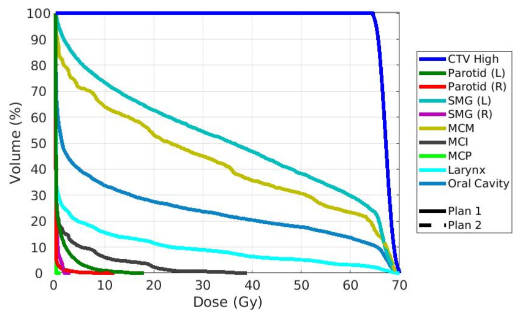
Figure 7.3: The dose distribution for different patients from Table 7.4. For Plan 1 the old weights w are used to get to an optimal solution $\hat{x}$. For Plan 2 the new weights $\hat{w}$ are used to get to an optimal solution $x^{\star}$. Part 1.



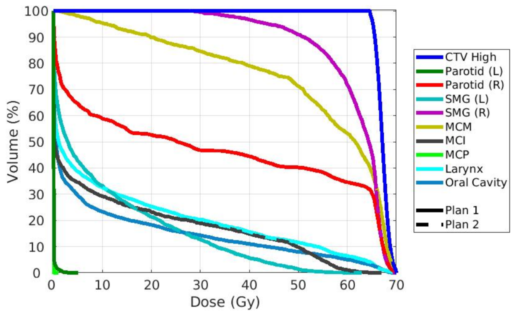
(a) Patient 9



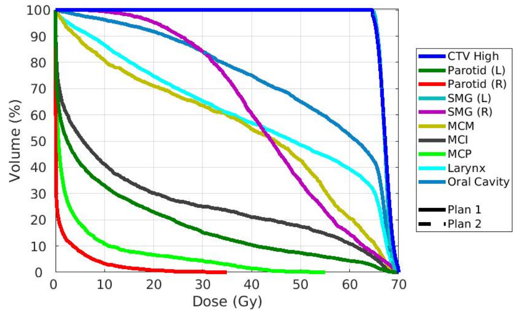
(b) Patient 10



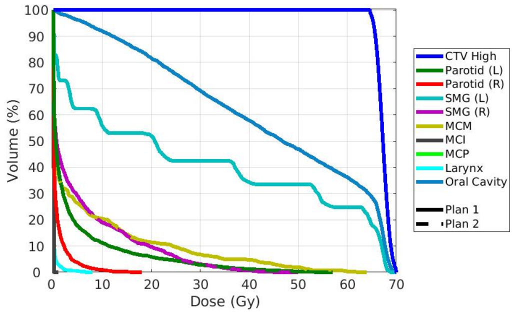
(c) Patient 11



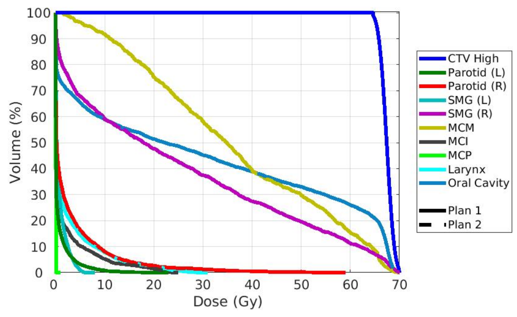
(d) Patient 12



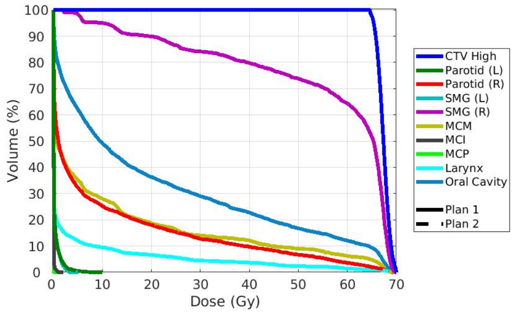
(e) Patient 13



(f) Patient 14

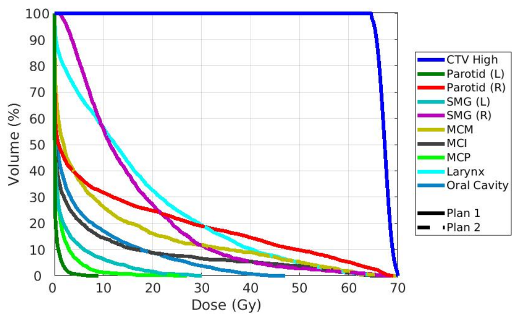


(g) Patient 15

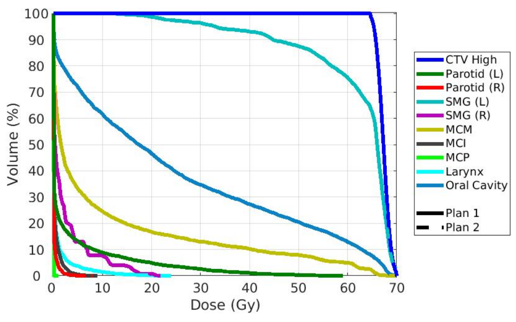


(h) Patient 16

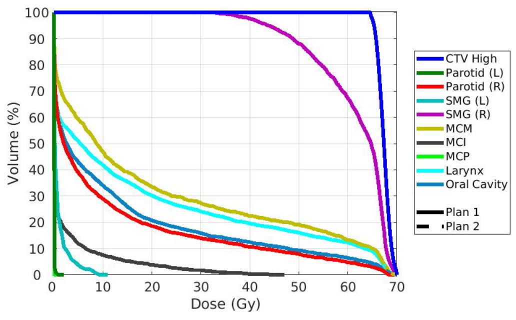
Figure 7.4: The dose distribution for different patients from Table 7.4. For Plan 1 the old weights w are used to get to an optimal solution $\hat{x}$. For Plan 2 the new weights $\hat{w}$ are used to get to an optimal solution $x^\star$. Part 2.



(a) Patient 17



(b) Patient 18



(c) Patient 19

Figure 7.5: The dose distribution for different patients from Table 7.4. For Plan 1 the old weights w are used to get to an optimal solution $\hat{x}$. For Plan 2 the new weights $\hat{w}$ are used to get to an optimal solution $x^{\star}$. Part 3.

8 CONCLUSION

The goal of this thesis was to identify the most relevant cost-functions for radiotherapy treatment planning with multicriteria optimisation.

Before we were able to identify the most relevant cost-functions, we had to find a method which could give us the most relevant cost-functions. We did this by assigning weights to the cost-functions. Before we could add new possible cost-functions we had to make it work for the cost-functions we already had with weights we already knew.

Unfortunately we could not get those weights back for non linear treatment plans, because there is too much noise to arrive at the exact treatment plan that we are looking for. This caused the new treatment plans to be worse than the old treatment plans, instead of the same or better.

For linear treatment plans we did get good results. We were able to obtain the same dose distribution as which we started with. To get this result we had to determine the dual of the optimisation problem and write in the form of an absolute duality gap minimization problem.

9 Discussion & recommendations

Firstly, we want to point out that the data that we used in Chapter 5 was of only one patient. We did look at the results for different patients, but that led to the same kind of results. That is why those results are not included in the report.

Secondly, the optimisation seems to work for almost all of the linear problems (Chapter 7). It went wrong if one OAR got the total weight. It could be a solution to remove this OAR from the problem when calculating the optimal weights $\hat{w}$. After optimising the weights, add the OAR again and give it weight zero. This is the same approach as we used for the case that $C\hat{x}$ had a value equal to zero.

Thirdly, the optimization works for the linear problems. Non-linear problems still can not be optimised, because for a non-linear problem there is too much noise to arrive at the exact treatment plan that we are looking for (Chapter 5). In practice not all of the automated treatment plans are linear. That is why for now we can not use this. I recommend to do more research to hopefully expand the possibilities to non-linear plans as well.

Fourthly, the end goal is to find the cost-functions for each OAR and the PTV that give the best plan. The best plan is a plan with for the OAR the goal to have a low dose in most of its total volume. For a PTV the best plan is a plan with a high dose in a big percentage of its total volume. When there is a method that finds the weights of non-linear functions, this method can be used to find the best cost-functions, by determining the weights of different cost-functions working on the same OAR or PTV.

A

Weights PTV and one or more OAR

Table A.1: The old weights w and the new weigths $\hat{w}$ (after inverse multicriteria optimisation) for the PTV and one OAR.

	w	$\hat{w}$
PTV	1.000	9.999E-1
Rectum (30)	2.634E-12	1.205E-4

Table A.2: The old weights w and the new weigths $\hat{w}$ (after inverse multicriteria optimisation) for the PTV and two OAR.

	w	$\hat{w}$
PTV	1.000	9.999E-1
Rectum (30)	2.634E-12	5.385E-6
Rectum (20)	3.349E-12	1.258E-4

Table A.3: The old weights w and the new weigths $\hat{w}$ (after inverse multicriteria optimisation) for the PTV and three OAR.

	w	$\hat{w}$
PTV	6.712E-01	6.722E-01
Rectum (30)	1.768E-12	1.505E-2
Rectum (20)	2.248E-12	8.589E-11
Rectum (10)	3.288E-1	3.127E-1

Table A.4: The old weights w and the new weigths $\hat{w}$ (after inverse multicriteria optimisation) for the PTV and four OAR.

	w	$\hat{w}$
PTV	6.648E-01	4.720E-04
Rectum (30)	1.751E-12	1.283E-10
Rectum (20)	2.227E-12	1.363E-12
Rectum (10)	3.257E-1	6.684E-1
External Ring 20 mm	9.482E-3	3.312E-1

Table A.5: The old weights w and the new weigths $\hat{w}$ (after inverse multicriteria optimisation) for the PTV and five OAR.

	w	$\hat{w}$
PTV	6.541E-1	5.823E-1
Rectum (30)	1.723E-12	1.479E-3
Rectum (20)	2.191E-12	7.642E-3
Rectum (10)	3.205E-1	1.805E-1
External Ring 20 mm	9.329E-3	1.688E-1
PTV Shell 5 mm	1.607E-2	5.921E-2

Table A.6: The old weights w and the new weigths $\hat{w}$ (after inverse multicriteria optimisation) for the PTV and six OAR.

	w	$\hat{w}$
PTV	5.883E-1	4.908E-1
Rectum (30)	1.549E-12	6.688E-16
Rectum (20)	1.970E-12	1.309E-2
Rectum (10)	2.882E-1	1.432E-1
External Ring 20 mm	8.391E-3	2.337E-1
PTV Shell 5 mm	1.445E-2	5.951E-2
Anus	1.006E-1	5.973E-2

Table A.7: The old weights w and the new weigths $\hat{w}$ (after inverse multicriteria optimisation) for the PTV and seven OAR.

	w	$\hat{w}$
PTV	5.834E-01	4.916E-01
Rectum (30)	1.536E-12	3.640E-19
Rectum (20)	1.954E-12	1.682E-2
Rectum (10)	2.858E-1	1.447E-1
External Ring 20 mm	8.321E-03	2.255E-1
PTV Shell 5 mm	1.433E-2	5.082E-2
Anus	9.977E-2	6.490E-2
PTV Shell 15 mm	8.361E-3	5.730E-3

Table A.8: The old weights w and the new weigths $\hat{w}$ (after inverse multicriteria optimisation) for the PTV and eight OAR.

	w	$\hat{w}$
PTV	5.791E-1	5.238E-1
Rectum (30)	1.525E-12	1.808E-16
Rectum (20)	1.939E-12	1.898E-2
Rectum (10)	2.837E-1	1.598E-1
External Ring 20 mm	8.258E-03	1.779E-1
PTV Shell 5 mm	1.423E-2	4.893E-2
Anus	9.902E-2	7.062E-2
PTV Shell 15 mm	8.299E-3	6.845E-16
PTV Shell 25 mm	7.468E-3	9.807E-6

Table A.9: The old weights w and the new weigths $\hat{w}$ (after inverse multicriteria optimisation) for the PTV and nine OAR.

	w	$\hat{w}$
PTV	5.791E-1	5.223E-1
Rectum (30)	1.525E-12	4.877E-12
Rectum (20)	1.939E-12	4.207E-2
Rectum (10)	2.837E-1	1.261E-1
External Ring 20 mm	8.258E-3	1.741E-1
PTV Shell 5 mm	1.423E-2	2.958E-2
Anus	9.902E-2	7.159E-2
PTV Shell 15 mm	8.299E-3	5.692E-3
PTV Shell 25 mm	7.468E-3	8.446E-3
Bladder	3.290E-12	2.016E-2

Table A.10: The old weights w and the new weigths $\hat{w}$ (after inverse multicriteria optimisation) for the PTV and ten OAR.

	w	$\hat{w}$
PTV	5.775E-1	4.731E-1
Rectum (30)	1.521E-12	1.050E-13
Rectum (20)	1.934E-12	3.562E-2
Rectum (10)	2.829E-1	1.132E-1
External Ring 20 mm	8.236E-3	1.957E-1
PTV Shell 5 mm	1.419E-2	2.760E-2
Anus	9.875E-2	6.685E-2
PTV Shell 15 mm	8.276E-3	1.859E-13
PTV Shell 25 mm	7.448E-3	4.209E-13
Bladder	3.282E-12	1.778E-2
Hip (L)	2.703E-3	7.018E-2

Table A.11: The old weights w and the new weigths $\hat{w}$ (after inverse multicriteria optimisation) for the PTV and eleven OAR.

	w	$\hat{w}$
PTV	5.749E-1	4.629E-1
Rectum (30)	1.514E-12	3.196E-12
Rectum (20)	1.926E-12	3.514E-2
Rectum (10)	2.817E-1	1.110E-1
External Ring 20 mm	8.200E-3	1.631E-1
PTV Shell 5 mm	1.412E-2	2.726E-2
Anus	9.832E-2	6.768E-2
PTV Shell 15 mm	8.240E-3	1.105E-12
PTV Shell 25 mm	7.415E-3	6.645E-13
Bladder	3.267E-12	1.609E-2
Hip (L)	2.691E-3	6.357E-2
Hip (R)	4.417E-3	5.331E-2

Table A.12: The old weights w and the new weigths $\hat{w}$ (after inverse multicriteria optimisation) for the PTV and twelve OAR.

	w	$\hat{w}$
PTV	5.000E-1	3.758E-1
Rectum (30)	1.317E-12	1.678E-19
Rectum (20)	1.675E-12	3.531E-2
Rectum (10)	2.449E-1	4.113E-2
External Ring 20 mm	7.131E-3	1.059E-2
PTV Shell 5 mm	1.228E-2	1.217E-19
Anus	8.550E-02	3.303E-2
PTV Shell 15 mm	7.166E-3	2.436E-19
PTV Shell 25 mm	6.449E-3	2.360E-19
Bladder	2.841E-12	1.254E-19
Hip (L)	2.340E-3	1.581E-2
Hip (R)	3.842E-3	7.963E-3
Patient	1.303E-1	4.804E-1

B
MATLAB CODE

B.1. NON-LINEAR PROBLEM

```
function [problem, data, solution] = make_EUD(problem,data)
%Changes the linear type cost-functions which compute maximum or minimum
     into a type 3 (gEUD) with parameter alpha equal to 30.

%Change all of the linear type cost-functions which compute maximum or
    minimum into a type 3 (gEUD) with parameter alpha equal to 30.
for k=1:length(problem)
    if problem(k).Type == 1 && size(data.matrix(problem(k).dataID).A,1)
        ~= 1 && ~strcmp(problem(k).Name,'Smoothing Linear')
        problem(k).Type = 3;
        problem(k).Parameters = 30;
    end
end

%Finds the new optimal solution
solution = primaldual(data.misc.size, data, problem, []);
end
```

```
function active_OAR_and_PTV (i_array)
%Converts the PTV to an objective function. Lets only the PTV and one
    OAR be active, with both weight 0.5

%Load a problem (problem_EUD), its data matrix (data) and optimal
    solution (x)
load /home/erasmusmc.nl/044469/yartos/data/Report/Prostate_VMAT_101_EUD

%Let all of the constraints and objectives be not active
for k=1:length(problem_EUD)
    problem_EUD(k).Active = 0;
end

%Convert the PTV to an objective cost−function with weight 0.5 and let
    it be active
problem_EUD(3).IsConstraint = 0;
problem_EUD(3).Active = 1;
problem_EUD(3).Weight = 0.5;

%Let one OAR be active with weight 0.5
problem_EUD(i_array).Weight = 0.5;
problem_EUD(i_array).Active = 1;

%Define a new name for this problem
problem_EUD_new = problem_EUD;

%Save the new problem (problem_EUD_new), the data matrix (data) and the
    optimal solution (x)
s = ['/home/erasmusmc.nl/044469/yartos/data/Report/2/' num2str(i_array)
    '/b/Prostate_VMAT_101_EUD_new'];
save(s, 'problem_EUD_new', 'x', 'data');
end
```

```
function multiple_active_OAR_and_PTV(i_array)
%Converts the PTV to an objective function. Lets only the PTV and
    i_array-10 OAR be active, with weight 0.5

%Load a problem (problem_EUD), its data matrix (data) and optimal
    solution (x)
load /home/erasmusmc.nl/044469/yartos/data/Report/Prostate_VMAT_101_EUD

%Let all of the constraints and objectives be not active
for k=1:length(problem_EUD)
    problem_EUD(k).Active = 0;
end

%Convert the PTV to an objective cost-function with weight 0.5 and let
    it be active
problem_EUD(3).IsConstraint = 0;
problem_EUD(3).Active = 1;
problem_EUD(3).Weight = 0.5;

%Let i_array - 10 OAR be active with weight 0.5
for i = 10:i_array
    problem_EUD(i).Weight = 0.5;
    problem_EUD(i).Active = 1;
end

%Define a new name for this problem
problem_EUD_new = problem_EUD;

%Save the new problem (problem_EUD_new), the data matrix (data) and the
    optimal solution (x)
s = ['/home/erasmusmc.nl/044469/yartos/data/Report/All_add/' num2str(
    i_array) '/b/Prostate_VMAT_101_EUD_new'];
save(s, 'problem_EUD_new', 'x', 'data');
end
```

```
function main_non_linear(i_array)
%Finds the optimal weights and corresponding optimal solution of a
    problem. Also calculates some extra information such as: the
    relative error of the weights, the relative error of the optimal
    solution and the goal function ||w^T*f(x)||^2_2  in different
    situations.

%Load a problem (problem_EUD_new), its data matrix (data) and optimal
    solution (x)
l =  ['/home/erasmusmc.nl/044469/yartos/data/Report/2/' num2str(i_array)
    '/a/Prostate_VMAT_101_EUD_new'];
load(l)

%Makes sure that the sum of the active objectives is equal to 1
problem_sum_weights = sum_weights(problem_EUD_new);

%Finds the new solution for the adjusted problem
options.OCond = 1e-8;
options.UseMatterhorn = 0;
x_new = primaldual(data.misc.size, data, problem_sum_weights, options);

%Finds new weights (inverse optimization)
[problem_new,weights_new, weights_old, sol, M] = find_weights(
    problem_sum_weights, data,x_new, 1);

%Calculates the relative error between the old and new weigths
rel_error_w = sum(abs(weights_new-weights_old))/size(weights_new,1);

%Finds the new solution for the new weights
x_new_weights = primaldual(data.misc.size, data, problem_new, options);

%Gives the new gradient matrix M with the new x
[M_new, ~] = find_M(problem_new, data, x_new_weights);

%Calculates the relative error between x and x_new_weights
rel_error_x = sum(abs(x_new-x_new_weights))/size(x_new,1);

%Calculates ||epsilon||
M_old_W_old = norm(M*weights_old,2);
M_old_W_new = norm(M*weights_new,2);
M_new_W_new = norm(M_new*weights_new,2);

%Saves the information found
```

```
s=['/home/erasmusmc.nl/044469/yartos/data/Report/2/' num2str(i_array) '/
    a/result.mat'];
save(s, 'problem_sum_weights','x_new','problem_new','weights_new','
    weights_old', 'sol', 'M','M_new','M_old_W_new','rel_error_w','
    x_new_weights','rel_error_x','data','M_old_W_old','M_new_W_new')
end
```

```
function [ problem ] = sum_weights(problem)
%This function equals the sum of the weights of the active objectives to
     one

%The array where the weights of the active objectives will be placed in
weights = [];

%Finds the weights of the active objectives
for k=1:length(problem)
    if problem(k).Active == 1
        weights = [weights, problem(k).Weight];
    end
end

%Equals the sum of the weights of the active objectives to one
weights = weights/sum(weights);

%Fives the problem the new weights
for k=1:length(problem)
    if problem(k).Active == 1
        problem(k).Weight = weights(1);
        weights = weights(2:end);
    end
end

end
```

B

```
function [ problem, weigths_obj, weights_old,sol,M ] = find_weights(
    problem, data, solution,sum_w)
%Optimizes the lagrangian: L(x,y) = w^T*f(x) + y^T*(g(x)-b) with f the
    objectives of a problem and g the constraints of the problem. The w'
    s are the weights assigned to each of the objectives. The function
    returns the problem with the new weights and the new weights.

for j=1:length(data.matrix)
    data.matrix(j).numvox = size(data.matrix(j).A, 1);
    data.matrix(j).A = single(full(data.matrix(j).A));
end

%The matrix where al the gradients of the objectives and constraints
    will be saved
M = [];

%Counts the amount of gradients
grad_count = [];

%Gets the gradients of all objectives and saves them in M.
for k=1:length(problem)
    if problem(k).IsConstraint == 0 && problem(k).Active == 1
        if problem(k).Type == 1
            [~, grad] = obj_linear(solution, data.matrix(problem(k).
                dataID), problem(k));
        elseif problem(k).Type == 2
            [~, grad] = obj_quadratic(solution, data.matrix(problem(k).
                dataID), problem(k));
        elseif problem(k).Type == 3
            [~, grad] = obj_eud(solution, data.matrix(problem(k).dataID)
                , problem(k));
        elseif problem(k).Type == 4
            [~, grad] = obj_ltcp(solution, data.matrix(problem(k).dataID
                ), problem(k));
        end
        M = [M, full(grad)];
        grad_count= [grad_count;size(grad,2)];
    end
end

%Amount of objectives
amt_w = length(grad_count);

```

```
%Length of a vector of gradients
amt_val = size(M,1);

%The matrix where all the values g(x) will be saved
g_val = [];

%Gets the gradients of all constraints and saves them in M.
for k=1:length(problem)
    if problem(k).IsConstraint == 1 && problem(k).Active == 1
         if problem(k).Type == 1
            [val, grad] = obj_linear(solution, data.matrix(problem(k).
                dataID), problem(k));
        elseif problem(k).Type == 2
            [val, grad] = obj_quadratic(solution, data.matrix(problem(k)
                .dataID), problem(k));
        elseif problem(k).Type == 3
            [val, grad] = obj_eud(solution, data.matrix(problem(k).
                dataID), problem(k));
        elseif problem(k).Type == 4
            [val, grad] = obj_ltcp(solution, data.matrix(problem(k).
                dataID), problem(k));
        end
        M = [M, full(grad)];
        grad_count= [grad_count;size(grad,2)];
        g_val = [g_val,val'-problem(k).Objective];
    end
end


%amount of constraints
amt_l = length(g_val);

%total amount of gradients (objectives and constraints)
amt_grad = size(M,2);

%Ensures y^T*(g(x)-b) is small (as close to zero as possible)
Aeq2 = [sparse(zeros(amt_l,amt_w)),sparse(double(diag(g_val))),sparse(
    zeros(amt_l,amt_val)),-speye(amt_l)];

%Ensures w^T*grad(f(x)) + y^T*grad(g(x)) is small (as close to zero as
    possible)
Aeq1= [sparse(double(M)),-speye(amt_val),sparse(zeros(amt_val,amt_l))];
sz_Aeq = size(Aeq1,2);
```

```

%Ensures that the sum of the weights of the objectives is equal to one
Aeq3 = [ones(1,amt_w) zeros(1,sz_Aeq - amt_w)];


%Constructs the matrix that takes our wishes into acount (see the above)
Aeq=[Aeq1;Aeq2;Aeq3];

%Constructs the matrix with the most perfect outcomes for our wishes
Beq=sparse(zeros(size(Aeq,1)-1,1));
Beq=[Beq;sum_w];

%The matrix indicates what we want to minimalise
H = sparse(amt_grad+1:sz_Aeq,amt_grad+1:sz_Aeq,ones(1,amt_val+amt_l),
    sz_Aeq, sz_Aeq);

%The leftbound of our solution (we want all the decision variables to be
     larger than zero)
LB = sparse(zeros(sz_Aeq,1));

 f=sparse(zeros(1,sz_Aeq));

%Gives the optimal weights for our problem
 opts=optimoptions('quadprog','Algorithm','interior-point-convex','
    Display','iter','OptimalityTolerance', 1e-18);
 [sol, ~, ~, ~, ~]=quadprog(H,f,[],[],Aeq,Beq,LB,[],[],opts);

%The new weights for the objectives
 weigths_obj = sol(1:amt_w);

%Saves the old weights
 weights_old = [];

%Assigns the new weights to the problem
 count = 1;
 for k=1:length(problem)
     if problem(k).IsConstraint == 0 && problem(k).Active == 1
         weights_old(count,1) = problem(k).Weight;
         problem(k).Weight = weigths_obj(count);
         count = count + 1;
     end
 end
 end
```

```
function [M, weights] = find_M(problem, data, solution)
%Finds the weights and the matrix M (the matrix with all of the
    gradients of the objectives and constraints)

for j=1:length(data.matrix)
    data.matrix(j).numvox = size(data.matrix(j).A, 1);
    data.matrix(j).A = single(full(data.matrix(j).A));
end

%The matrix where al the gradients of the objectives and constraints
    will be saved
M = [];
weights= [];
%Counts the amount of gradients
grad_count = [];

%Gets the gradients of all objectives and saves them in M.
for k=1:length(problem)
    if problem(k).IsConstraint == 0 && problem(k).Active == 1
        if problem(k).Type == 1
            [~, grad] = obj_linear(solution, data.matrix(problem(k).
                dataID), problem(k));
        elseif problem(k).Type == 2
            [~, grad] = obj_quadratic(solution, data.matrix(problem(k).
                dataID), problem(k));
        elseif problem(k).Type == 3
            [~, grad] = obj_eud(solution, data.matrix(problem(k).dataID)
                , problem(k));
        elseif problem(k).Type == 4
            [~, grad] = obj_ltcp(solution, data.matrix(problem(k).dataID
                ), problem(k));
        end
        M = [M, full(grad)];
        grad_count= [grad_count;size(grad,2)];
        weights = [weights; problem(k).Weight];
    end
end

%Amount of objectives
amt_w = length(grad_count);

%Length of a vector of gradients
amt_val = size(M,1);
```

```
%The matrix where all the values g(x) will be saved
g_val = [];

%Gets the gradients of all constraints and saves them in M.
for k=1:length(problem)
    if problem(k).IsConstraint == 1 && problem(k).Active == 1
        if problem(k).Type == 1
            [val, grad] = obj_linear(solution, data.matrix(problem(k).
                dataID), problem(k));
        elseif problem(k).Type == 2
            [val, grad] = obj_quadratic(solution, data.matrix(problem(k)
                .dataID), problem(k));
        elseif problem(k).Type == 3
            [val, grad] = obj_eud(solution, data.matrix(problem(k).
                dataID), problem(k));
        elseif problem(k).Type == 4
            [val, grad] = obj_ltcp(solution, data.matrix(problem(k).
                dataID), problem(k));
        end
        M = [M, full(grad)];
        grad_count= [grad_count;size(grad,2)];
        g_val = [g_val,val'-problem(k).Objective];
        weights = problem(k).Weight;
    end
end
end
```

B.2. LINEAR PROBLEM

```
function main_linear( i_array )
%Finds the optimal weights (alpha) and corresponding optimal solution (x
    ) of a problem.

l =  ['/home/erasmusmc.nl/044469/yartos/data/Report/Proton/Data/Protons_
    ' num2str(i_array) '.mat'];
load(l)

%Find the optimal solution for the problem, part one.
[ CMatrix, b, A,Objectives,alpha] = linear_problem_one(problem,data);

% Define alpha equal to 1 for every objective. Or do not define alpha
    here, then the vector alpha contains the original weights.
alpha = [ones(1,size(CMatrix,1))]';
alpha=alpha/sum(alpha);

%Find the optimal solution for the problem, part two.
[ CMatrix, x, b, alpha, A,Objectives,p] = linear_problem_two(alpha,
    CMatrix, b, A,Objectives);

%Create a new problem that is the old problem without the constraints
    for which Cx (CMatrix*x) is equal to zero. Leave out to see what
    happens if we keep these constraints.
test = CMatrix * x;
CMatrix_nz = [];
Objectives_nz = [];
for i=1:size(CMatrix,1)
    if test(i) ~= 0
        CMatrix_nz = [CMatrix_nz; CMatrix(i,:)];
        Objectives_nz = [Objectives_nz, Objectives(i)];
    end
end

%find the optimal weights alpha
[ alpha_new, p_new ] = find_alpha( CMatrix_nz, x, b, A);

%Copy the problem to a new problem
problem_new = problem;

%Assign the new weights to the corresponding objectives of the problem
for i = 1:length(Objectives_nz)
    problem_new(Objectives_nz(i)).Weight = alpha_new(i);
```

```
end

%If Cx (CMatrix*x) is equal to zero for a constraint the weight of the
    constraint is zero
for i=1:size(CMatrix,1)
    if test(i) == 0
        alpha(i) = 0;
        alpha_new = [alpha_new(1:i-1);0;alpha_new(i:end)];
    end
end

%This function equals the sum of the weights of the objectives to one
alpha=alpha/sum(alpha);

%Find the new optimal solution
[ ~, x_new, ~, ~,~,~] = linear_problem(problem_new,data,alpha_new);

%Let all of the constraints and objectives be not active
for i=1:length(problem_new)
    problem_new(i).Active=0;
end

%Let all of the constraints of our original problem be active
for i = 1:length(Objectives)
    problem_new(Objectives(i)).Active = 1;
end

%Calculate relative errors
rel_error_x = sum(abs(x-x_new))/size(x_new,1);
rel_error_alpha = sum(abs(alpha-alpha_new))/size(alpha,1);

%Calculate our goal-function
opt_x_alpha = alpha.'*CMatrix*x - b.'*p;
opt_x_alpha_new = alpha_new.'*CMatrix*x - b.'*p_new;
opt_x_new_alpha_new = alpha_new.'*CMatrix*x_new - b.'*p_new;

%Saves the information found
s=['/home/erasmusmc.nl/044469/yartos/data/Report/Proton/Results/Protons_
    ' num2str(i_array) '_result.mat'];
save(s, 'problem','data', 'CMatrix', 'p','opt_x_alpha','opt_x_alpha_new'
    , 'opt_x_new_alpha_new','x', 'b', 'alpha','A', 'p_new','alpha_new',
    'problem_new', 'x_new', 'CMatrix_nz', 'Objectives', 'Objectives_nz',
    'rel_error_x','rel_error_alpha')
```

```
end
```

```
function [ CMatrix, x, b, alpha, A,Objectives] = linear_problem(problem,
    data,alpha)
% Constructs a linear radiotherapy problem with its weights (alpha) and
% finds its optimal solution (x)

% New optimization problem

% Constrain dose to CTV High between 64.68 and 69.96
CTVHighMatrix = data.matrix(1).A;

% Objectives: minimise mean Parotids, SMGs SCM, MCM, MCI, MCP
% Since these plans are optimised robustly, only take the 1st row
Objectives = [];
CMatrix = [];
for j=1:length(problem)
    if problem(j).Active && ~problem(j).IsConstraint
        if strmatch(problem(j).Name, {'Parotid (L) (mean)', 'Parotid (R)
            (mean)', 'SMG (L) (mean)', 'SMG (R) (mean)', 'SCM (mean)',
            'MCM (mean)', 'MCI (mean)', 'MCP (mean)', 'Larynx (mean)', '
            Oral Cavity (mean)'})
            Objectives = [Objectives j];
            ObjIdx = length(Objectives);
            fprintf('Matched %s as objective %d\n', problem(j).Name,
                ObjIdx);
            CMatrix(ObjIdx, :) = data.matrix(problem(j).dataID).A(1, :);
        end
    end
end

% Finding the optimal solution

% We define Ax>b with -Ax<-b
A = [CTVHighMatrix; ...
    -CTVHighMatrix];
b = [ones(size(CTVHighMatrix, 1), 1) * 64.68; ...
    -ones(size(CTVHighMatrix, 1), 1) * 69.96];

%Defines weights' times Cx
f = alpha'*CMatrix;

NumEl = size(A, 2);

```

```
%Finds the optimal solution:
 options = optimoptions('linprog','OptimalityTolerance',1e-10);
[x,~,~,~,temp] = linprog(f, -A, -b, [], [], zeros(NumEl, 1),[], options)
    ;

%Makes sure that there are no negative x's. If there are negative x's we
%change it in zero
for i =1:length(x)
     if x(i) < 0
         x(i) = 0;
     end
end


%Saves the dual solution
p = temp.ineqlin;

end
```

```
function [ CMatrix, b, A,Objectives,alpha] = linear_problem_one(problem,
    data)
% If we want to define the weights alpha ourselves, we need to split the
% linear_problem function into two parts. This is the first part.

% New optimization problem

% Constrain dose to CTV High between 64.68 and 69.96
CTVHighMatrix = data.matrix(1).A;

% Objectives: minimise mean Parotids, SMGs SCM, MCM, MCI, MCP
% Since these plans are optimised robustly, only take the 1st row
Objectives = [];
CMatrix = [];
alpha = [];
for j=1:length(problem)
     if problem(j).Active && ~problem(j).IsConstraint
          if strmatch(problem(j).Name, {'Parotid (L) (mean)', 'Parotid (R)
              (mean)', 'SMG (L) (mean)', 'SMG (R) (mean)', 'SCM (mean)',
             'MCM (mean)', 'MCI (mean)', 'MCP (mean)', 'Larynx (mean)', '
             Oral Cavity (mean)'})
              Objectives = [Objectives j];
              alpha = [alpha; problem(j).Weight];
              ObjIdx = length(Objectives);
              fprintf('Matched %s as objective %d\n', problem(j).Name,
                    ObjIdx);
                CMatrix(ObjIdx, :) = data.matrix(problem(j).dataID).A(1, :);
            end
        end
    end


% Finding the optimal solution

% We define Ax>b with -Ax<-b
A = [CTVHighMatrix; ...
    -CTVHighMatrix];
b = [ones(size(CTVHighMatrix, 1), 1) * 64.68; ...
    -ones(size(CTVHighMatrix, 1), 1) * 69.96];

end
```

```
function [ CMatrix, x, b, alpha, A,Objectives,p] = linear_problem_two(
    alpha, CMatrix, b, A,Objectives)
% If we want to define the weights alpha ourselves, we need to split the
% linear_problem function into two parts. This is the second part.

%Define weights' times Cx
f = alpha'*CMatrix;

NumEl = size(A, 2);

%Finds the optimal solution:
 options = optimoptions('linprog','OptimalityTolerance',1e-10);
[x,~,~,~,temp] = linprog(f, -A, -b, [], [], zeros(NumEl, 1),[], options)
    ;

%Makes sure that there are no negative x's. If there are negative x's we
%change it in zero.
for i =1:length(x)
    if x(i) < 0
        x(i) = 0;
    end
end

%Saves the dual solution
p = temp.ineqlin;
end
```

```
function [ alpha_new,p_new ] = find_alpha( CMatrix, x, b, A )
%Finds the optimal weigths (alpha) for the problem CMatrix.

%alpha'*C*x-eps = 0
Aeq1 = [(CMatrix*x)', zeros(1,size(b,1)), -1,0];

%b'*p + gamma = 0
Aeq2 = [zeros(1,size(CMatrix*x,1)), -b',0,-1];

%sum (alpha) = 1
Aeq3 = [ones(1,size(CMatrix*x,1)),zeros(1,size(b,1)),0,0];

Aeq = [Aeq1;Aeq2;Aeq3];
beq = [zeros(size(Aeq,1)-1,1);1];

%-C'alpha + A'p < 0
MA =[-CMatrix', A', zeros(size(A,2),2)];
Mb = zeros(size(MA,1),1);

%gamma + epsilon
getey = [zeros(1,size(Aeq,2)-2),1,1]';

%The weights can not be smaller than zero
lb = [zeros(1,size(Aeq, 2)-2) -Inf -Inf];

%Calculate the optimal solution
options = optimoptions('linprog','Algorithm','dual-simplex', 'Display',
    'iter', 'OptimalityTolerance', 1e-9, 'ConstraintTolerance', 1e-9);
temp=linprog(getey, MA, Mb, Aeq, beq, lb,[],options);

%Gets the new optimal solution and the new dual solution
alpha_new = temp(1:size(CMatrix*x,1))/sum(temp(1:size(CMatrix*x,1)));
p_new = temp(size(CMatrix*x,1)+1:size(CMatrix*x,1)+size(b,1));
end
```

C

Bibliography

REFERENCES

[1] S. Breedveld, *Radiotherapy,* (2013), [Online; accessed June 16, 2019].

[2] R. Van Haveren, *Lexicographic Reference Point Method for Automatic Treatment Planning in Radiation Therapy*, Master's thesis, Delft University of technology, the Netherlands (2014).

[3] S. Breedveld, B. van den Berg, and B. Heijmen, *An interior-point implementation developed and tuned for radiation therapy treatment planning,* Computational Optimization and Applications **68**, 209 (2017).

[4] S. Breedveld, D. Craft, R. van Haveren, and B. Heijmen, *Multi-criteria optimization and decision-making in radiotherapy,* European Journal of Operational Research **277**, 1 (2019).

[5] S. Breedveld and B. Heijmen, *Data for trots – the radiotherapy optimisation test set,* Data in Brief **12**, 143 (2017).

[6] S. Breedveld, P. R. M. Storchi, and B. J. M. Heijmen, *The equivalence of multi-criteria methods for radiotherapy plan optimization,* Physics in Medicine and Biology **54**, 7199 (2009).

[7] A. Babier, J. J. Boutilier, M. B. Sharpe, A. L. McNiven, and T. C. Y. Chan, *Inverse optimization of objective function weights for treatment planning using clinical dose-volume histograms,* Physics in Medicine & Biology **63**, 105004 (2018).