COUPLED REDUCED ORDER MODEL AND ADJOINT METHODOLOGIES FOR PROTON THERAPY

(DOUBLE) BACHELOR THESIS

by

Lars kleyn Winkel

in partial fulfillment of the requirements for the degree of

Bachelor of Science
in Applied Physics and Applied Mathematics
at the Delft University of Technology.

Delft, October 6, 2019

Supervisors:	Dr. Z. Perkó	Applied Physics
	Dr. ir. D. Lathouwers	Applied Physics
	Dr. ir. M. van Gijzen	Applied Mathematics
Thesis committee:	Prof. dr. ir. A.W. Heemink,	Applied Mathematics

An electronic version of this thesis is available at http://repository.tudelft.nl/.

REACTOR INSTITUTE DELFT

TU Delft Delft University of Technology

MP&T Medical Physics & Technology

ABSTRACT

The goal of this project is to see if we can improve the treatment plan for proton therapy by using reduced order models and adjoint theory for proton therapy. We shall use a singular value decomposition on a dose distribution matrix to obtain the modes from which we can reconstruct every dose distribution. Using adjoint methodologies for proton therapy, we will define a response from which we can find the sensitivities, or gradient, in order to fit a Hermite interpolation polynomial on multidimensional simplices.

The results show that the Hermite interpolation polynomial is a useful tool to find responses for low dimensional problems. For errors in one or two dimensions, the Hermite polynomial was able to reconstruct all dose distributions with $R^2 = 1$. However, for an error in three dimensions the Hermite polynomial sometimes fails to reconstruct the dose distribution to within acceptable margins.

We conclude that the combination of a ROM and adjoint method to find Hermite interpolation polynomial is a promising tool in order to further improve the proton therapy treatment plan. Further research should be done in order to determine whether Hermite interpolation can be used in every scenario, or if it fails if the grid becomes irregular. Finally, the extrapolating qualities of the Hermite polynomial should be tested.

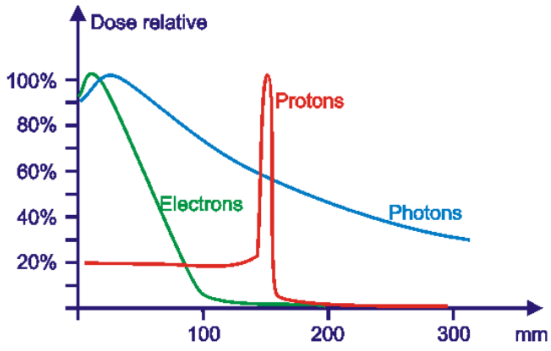
Contents

1

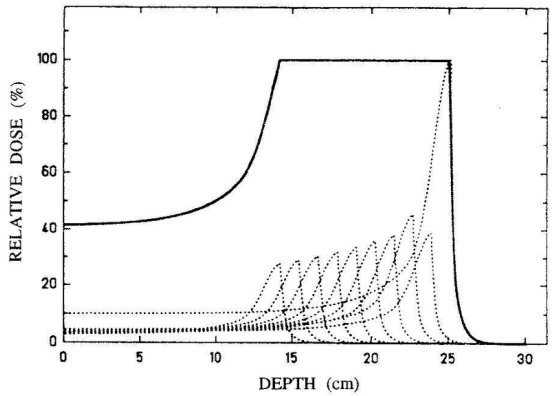
INTRODUCTION

1.1. PROTON THERAPY

A promising method of radio therapy for the treatment of cancer patients is proton therapy. Proton therapy is an alternative for the more widely used photon therapy (Van Galen [1], Kasbergen [2], Paganetti [3]). The fundamental difference between the two methods is that photon therapy uses high energy light particles to radiate a tumor, whilst proton therapy uses charged particles, protons, for radiation of the tumor. Since photon therapy uses light particles, their energy deposition can be modeled by an exponential. From this we can deduce that the dose in the patient before the tumor is higher that the dose in the actual tumor itself, meaning the tissue in front of the tumor is damaged more than the actual tumor. However, the proton energy distribution does not show a peak before the tumor. The peak can be positioned in the tumor itself. Figure 1.1a shows a comparison between the dose deposition between the two techniques.



(a) Comparison of proton, electron and photon energy deposition. Source: (Dang [4]).



(b) Figure shows how a Spread Out Bragg Peak can be constructed as a superposition of individual pristine peaks. The weights have to chosen carefully such that the iconic SOBP plateau arrises. Source: (Davino [5]).

Figure 1.1: Comparison of photon therapy to proton therapy.

The curve that represent the energy deposition by ionizing radiation is called the Bragg peak (Charlie Ma and Lomax [6]), named after William Bragg. Figure 1.1a shows the Bragg curve for different kinds of ionizing radiation. The Bragg curve for monochromatic proton beam is called a pristine peak. Different proton energies give shifted pristine peaks. The peaks show where the dose is deposited in the patient. Adding these different pristine peak energy depositions, call them ϕ_i's, with carefully chosen weights, say w_i's, we can reconstruct the Spread Out Bragg Peak (SOPB) from Figure 1.1b, $SOBP(x) = \sum_i w_i \phi_i(x)$, as a superposition of these pristine peaks.

From figure 1.1a we see that proton therapy is more 'precise' than photon therapy. Not only does it deliver most of its dose in the desired region, there is also no exit dose. The latter meaning at a certain depth within the patient the free proton density is nearly zero. However, this brings additional challenges: since a higher dose can be administered at once, a slight overshoot can badly damage any nearby organs. It is thus critical that we know how changes in the patient's body or discrepancies in patient positioning affect the dose distribution within the patient, in order to ensure no unnecessary organ damage.

1.2. PROJECT GOAL

The optimal treatment plan set up for a patient is called the nominal scenario. When a patient receives treatment, it is very unlikely he or she will be in the exact same position, or state, as the nominal scenario. We will refer to any such scenario that is not the nominal case as an error scenario. Since we will find the dose distribution numerically, the entire dose distribution for a single error scenario will be contained in a vector $\mathbf{d}_i$. The dose distribution vectors for all scenarios will be stored in the *dose distribution matrix*: $D = [\mathbf{d}_1 \mathbf{d}_2 ... \mathbf{d}_n]$.

Using a reduced order model on the dose distribution matrix, as described by Van Galen (Van Galen [1, p. 5-6]), we are able to reduce the order of the problem, meaning the problem is easier to handle physically and mathematically, as well as to find the modes from which the signal is build up.

Since these modes do make up all the column vectors of the matrix, we are able to reconstruct every dose distribution for any error scenario to within any desired precision. We now state that for all error scenarios, also the ones that are not generated, the dose distribution consists of exactly these modes. The question that remains, is how to add up these modes in order to restore the dose distribution, which is the topic of this project: *What are the coefficients for the modes as a function of the error scenario?*

A great part of this question has been covered by J.W. Van Galen in his thesis (Van Galen [1]). Van Galen used regression to find a function that fits the data points. The conclusion yields that regression of polynomials on the first few modes works well, but fails for the higher order modes. To perform a better fit, more information is needed. This information comes from the adjoint flux, by calculating the sensitivities as Kasbergen describes in his thesis (Kasbergen [2]). Basically, by finding the adjoint flux we mean solving an equation that is closely related to the original equation, but now provides more information. Kasbergen derived the sensitivities needed in this project. These sensitivities will then be used to fit a polynomial with corresponding derivatives and function values. The polynomial will be fitted using Hermite interpolation.

A one-dimensional model will be used in this project to model a patient. The model solves the one-dimensional Fokker-Planck approximation to the Boltzmann transport (Uilkema [7]) equation. The code that implements this model is written in Fortran. The code uses a CT-scan as input and outputs the dose distribution by solving the Fokker-Planck approximation of the Boltzmann transport equation. The sensitivities, or gradient, come from the code by calculating the inner products described by Kasbergen (Kasbergen [2, p. 20-22]).

The structure of the report is as follows: the relevant theory will be presented in chapter 2. The methods used will be discussed in chapter 3 followed by the results and discussion in chapter 4. The report will be concluded by the the conclusion in chapter 5. The project is part of the Bachelor Applied Physics and Applied Mathematics at Delft University of Technology.

2

THEORY

2.1. PROTON THERAPY TREATMENT PLANNING

As mentioned in Section 1.1, proton therapy is a promising alternative to photon therapy. Due to its high precision, a much higher dose can be administered to a tumor at once. However, a slight discrepancy in the patient can cause a high dose deposition in a healthy tissue. To monitor how the dose deposition in the patient will look like, we need a function that predicts the dose distribution from a CT-scan, such that the treatment plan can be updated such that a patient always gets an optimal treatment. Using the CT-scan as input, the Fortran code that represents the model will find the stopping power, which according to Uilkema [7, p. 26] represent the "energy transfer of the incident proton to the atomic electrons" for the protons, which is necessary to find the total dose distribution. Other important input parameters are the cosine scatter angle μ (Figure 2.1), which represents the direction of the protons, and the total macroscopic scatter cross section σ_t in cm^{-1}. The total macroscopic scatter cross section is a measure for how many protons are absorbed by the medium, which in practice is very low.

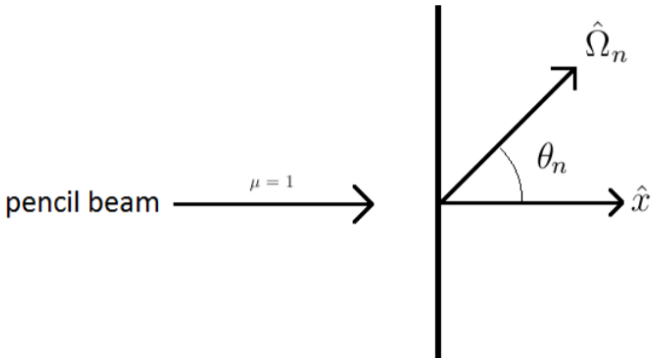


Figure 2.1: Relation between $\hat{x}$,$\hat{\Omega}$ and θ with $\mu = \cos(\theta)$. Uilkema [7].

2.1.1. BOLTZMAN TRANSPORT EQUATION

In his master thesis, Uilkema derives the one-dimensional Fokker-Planck approximation of the linear one-dimensional Boltzmann equation (Uilkema [7, p. 21-30]):

$$\mu \frac{\partial \phi(x,E,\hat{\Omega})}{\partial x} + \sigma_t \phi(x,E,\hat{\Omega}) = \frac{\partial (S(E)\phi(x,E,\hat{\Omega}))}{\partial E} \tag{2.1}$$

$\phi(x,E,\hat{\Omega})$ is the proton flux at position x with energy E and direction $\hat{\Omega}$, μ is the cosine scatter angle, σ_t is the total macroscopic scatter cross section and $S(E)$ the stopping power at energy E. This equation is solved for the proton flux ϕ numerically by the Fortran code (Lathouwers [8]). The code uses a Galerkin based finite element method in order to conserve energy (Kuzmin [9]). The solution of the partial differential equation, the proton flux, is used to determine the total administered dose:

$$Dose = \iiint \frac{[S(E)\phi(x,E,\hat{\Omega})]}{\rho} dx dE d\hat{\Omega} \tag{2.2}$$

with ρ the mass density.

2.1.2. PARAMETER ERRORS

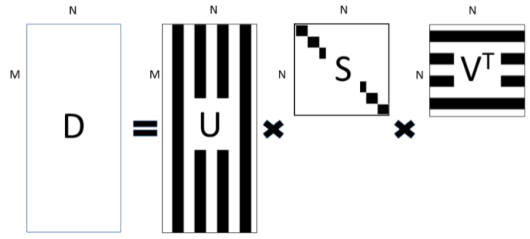
Parameters that can be perturbed in equation 2.1 are the stopping power $S(E)$ and the total macroscopic scatter cross section σ_t. Perturbations in the stopping power can arise from organ movement, weight gained or lost by the patient, a different composition of the surrounding tissue, to name just a few.

Since these parameters are directly involved in the partial differential equation that determines the proton flux, they directly influence the dose distribution. In order to find a function that relates the error in the patient to the dose distribution, the effects of the errors on the dose distribution have to be modeled for several error scenarios.

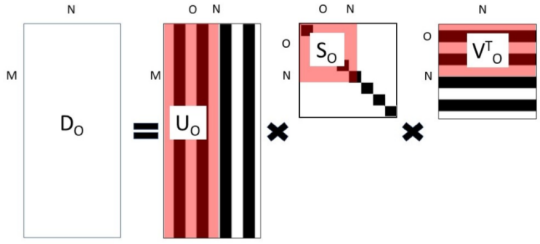
2.2. REDUCED ORDER MODELLING

The Fortran code that solves the Boltzmann equation outputs the coefficients for the Galerkin finite element method. A Finite Element Method, such as Galerkin's, solves a partial differential equation. A FEM multiplies the partial differential equation with a weighted sum basis functions, where the weights, or coefficients, need to be calculated in order to approximate the solution ([9]). These coefficients are simply the values of the dose deposition in the boundary of each element, which is not the same as the proton flux (see Equation 2.2). Since every element has two boundaries, we have twice as many coefficients as elements (that is, two on every boundary with exception of the first and last). Since these coefficients form a finite set of numbers, we arrange them in a vector: **d**.

As mentioned in Section 2.1.2, several error scenarios are to be modeled. Each scenario yields such a vector **d**. If we have k elements in our finite element method, and model N error scenarios, we can arrange the dose distributions in a $M \times N$ matrix, with $M = 2k$, which we shall call the dose distribution matrix: $D = [\mathbf{d}_1\mathbf{d}_2...\mathbf{d}_n]$. As Van Galen described in his thesis (Van Galen [1]), an effective method for reducing the dimensionality of the given problem is doing a thin singular value decomposition (SVD) on the dose distribution matrix. Thin in this case means the columns of U and V are normilised, by dividing the singular val Since the dose distriue by the corresponding normilisabution matrtion conix is not necessarily square, an eigenvalue decomposition is not an option.



(a) Figure shows a graphical representation of the SVD. U and V contain orthogonal vectors and S is the diagonal matrix containing the singular values. Source: Van Galen [1, p. 6].



(b) Figure shows a graphical representation of a truncated SVD. Subscript o denotes the first 'o' orders are used to construct the doses distribution matrix. Source: Van Galen [1, p. 6]

Figure 2.2: Illustration of how the singular value decomposition reduces the order of the problem.

The singular value decomposition theorem states that any matrix of any size can be decomposed into three matrices accordingly: $D = USV^*$. The matrices U and V contain orthonormal vectors and are called the left- and right-singular-vectors. The matrix S is a diagonal matrix containing the singular values (in decreasing order). The non-zero entries of the S matrix are the square roots of the non-zero eigenvalues eigenvalues of the D^*D and DD^* matrices. A visual representation of the SVD decomposition is given by the following figure:

By excluding some (small) singular values, the original matrix can be reconstructed to any precision required in a voxel by voxel comparison. The exclusion of some singular values means the sizes of U,S and V change from $M \times N$, $N \times N$ and $N \times N$ respectively to $M \times K$, $K \times K$ and $N \times K$ respectively. This does not change the size of the constructed matrix, which is still $M \times N$. The truncated SVD is represented graphically in figure 2.2b.

2.2.1. LEFT SINGULAR VECTORS

The left singular vectors, contained in the U matrix, in combination with the corresponding singular values of matrix S are the so called *modes* of the original dose distribution. If we define $D_m = US$, then the column vectors of D_m are the modes from which every dose distribution can be reconstructed. Thus we can write every dose distribution as a linear combination of the modes as follows:

$$\mathbf{d}_j = \sum_{i=1}^{N} w_{ij} \mathbf{D}_m^i \tag{2.3}$$

where w_{ij} is the weight (not the same as in section 1.1) and a function of the error scenario, and D_m^i is the i'th column vector (or mode) from the matrix D_m.

2.2.2. RIGHT SINGULAR VECTORS

The right singular vectors contain the weights w_{ij}, or coefficients, mentioned in Section 2.2.1 for the superposition of the modes to reconstruct a certain dose distribution. In this project we want to find a function that relates the error scenario to the corresponding weights in the right singular vectors. The low order right singular vectors are rather simple, such that they can be approximated by polynomials, yet the higher order right singular vectors become oscillatory and thus harder to fit low order polynomials. The oscillatory behaviour can be explained by the fact that the right (and left) singular vectors constitute a set of orthonormal vectors. In order to sustain the orthonormality of the singular vectors, higher order vectors become oscillatory while the low order vectors are relatively simply shaped.

2.3. ADJOINT THEORY

In proton therapy, we are often dealing with calculating inner products of the proton flux with an operator over space, energy and direction (dx,dE,$d\Omega$). We denote this inner product with the following notation:

$$< A\phi, \psi >= \iiint (A\phi)\psi \, dx dE d\hat{\Omega} \tag{2.4}$$

with A an operator and ϕ the proton flux. An *adjoint* state of the operator A, the adjoint operator, is defined as follows: $< A\phi, \psi >=< \phi, A^\dagger \psi >$ where $A^\dagger$ is the adjoint of A. The adjoint of a matrix is its transpose, the adjoint of the gradient ∇ is $-\nabla$ and the adjoint of a constant is the constant itself.

2.3.1. RESPONSE CALCULATION

A response is what we define to be an inner product of the following kind: $R =< f, \phi >$ with f a function and ϕ the proton flux. The dose is a response with the stopping power $S(E)$ over density $\rho(x)$ as a function operator: $Dose =< S(E)/\rho(x), \phi) >$. An error scenario will affect the stopping power, density and the parameters in the Boltzmann equation thus changing the value of the dose response. Recalculating the dose with the new stopping power and proton flux is computationally expensive.

Adjoint theory can provide us with a method to efficiently calculate the change in response. As shown by Kasbergen in his thesis (Kasbergen [2]) and by Abdel-Khalik *et al.* [10], we can calculate the *change* in response by the following important equation:

$$\Delta R = - < \phi_0^\dagger, \Delta A \phi_0 > \left(= \iiint \phi_0^\dagger (\Delta A \phi_0) dx dE d\hat{\Omega} \right) \tag{2.5}$$

With ΔR the change in response, A an operator, ϕ_0 the (forward) proton flux and $\phi_0^\dagger$ the *adjoint proton flux*. The adjoint proton flux is obtained by solving the adjoint Boltzmann transport equation:

$$-\mu \frac{\partial \phi^\dagger(x,E,\hat{\Omega})}{\partial x} + \sigma_t \phi^\dagger(x,E,\hat{\Omega}) = -S(E) \frac{\partial (\phi^\dagger(x,E,\hat{\Omega}))}{\partial E} \tag{2.6}$$

The adjoint Boltzmann transport equation from Equation 2.6 is derived by Kasbergen in his thesis (Kasbergen [2, p. 17]). Since we obtain the change in response, we need a reference scenario R_0 such that $R = R_0 + \Delta R$. The reference scenario R_0 is the nominal scenario mentioned in Section 1.2, which we denote by A_0 and ϕ_0. Since we use the nominal scenario in the calculation of the response, we only need to calculate the forward and adjoint flux once. With the change in the operator ΔA we can then find the change in response with Equation 2.5.

2.3.2. SENSITIVITY CALCULATION

Adjoint theory can also supply sensitivities with the respect to an arbitrary parameter p_j with the help of equation 2.5. As explained by Abdel-Khalik *et al.* [10] we can simply divide Equation 2.5 by the change in the parameter Δp_j, and then take the limit as $\Delta p_j \to 0$. The main results is described by Equation 2.7:

$$\frac{\partial R}{\partial p_j} = - < \phi_0^\dagger, \left(\frac{\partial A}{\partial p_j} \right) \phi_0 > \tag{2.7}$$

In other words: we can find the value of any response, as long as it can be written as an inner product over space, energy and direction. We can find the full gradient of every response with respect to any parameter by using the nominal forward and adjoint flux, and the derivative of the operator with respect to the parameters. In terms of the project goal, we can find the value of any response (in $\mathbb{R}^n$, see Section 2.4) and the full gradient in the response point.

2.4. HERMITE INTERPOLATION

In order to use the extra information obtained with the full gradient from the adjoint analysis, we will use the Hermite interpolation method (Vuik *et al.* [11]). This method uses the given gradient, $\mathbf{G}(\mathbf{x})$, to fit a polynomial $P(\mathbf{x})$ such that $\nabla P(\mathbf{x}) = \mathbf{G}(\mathbf{x})$ in the snapshot points on which we will fit our function. The snapshot points will be the points on which we will fit our Hermite polynomial. The full gradient and responses will be obtained from the adjoint analysis. We will thus fit a function $P(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ with n the number of errors.

2.4.1. ONE-DIMENSIONAL HERMITE INTERPOLATION

When two data points are known, say $(x_0, f(x_0))$ and $(x_1, f(x_1))$ and their respective derivatives are $(x_0, f'(x_0))$ and $(x_1, f'(x_1))$, a third order polynomial can be fitted through the data points in the following way:

1. Write $P_3(x) = ax^3 + bx^2 + cx + d$, and $P_3'(x) = 3ax^2 + 2bx + c$.
2. Equate $P_3(x_i) = f(x_i)$ and $P_3'(x_i) = f'(x_i)$, for $i = 0, 1$ yielding a system of four equations with four unknowns.
3. Solve the system, obtaining a, b, c, d and thus solving for the polynomial.

Since we obtain a system of four equations with four unknowns we can write the equations in matrix form (Vuik *et al.* [11, p. 19]):

$$\begin{bmatrix} 1 & x_0 & x_0^2 & x_0^3 \\ 1 & x_1 & x_1^2 & x_1^3 \\ 0 & 1 & 2x_0 & 3x_0^2 \\ 0 & 1 & 2x_1 & 3x_1^2 \end{bmatrix} \begin{bmatrix} d \\ c \\ b \\ a \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f(x_1) \\ f'(x_0) \\ f'(x_1) \end{bmatrix} \tag{2.8}$$

Any interpolation polynomial based on function values as well as derivatives, or sensitivities, is called an *Hermite Interpolation Polynomial*. An example of how Hermite interpolation can improve a fit is given in figure 2.3.

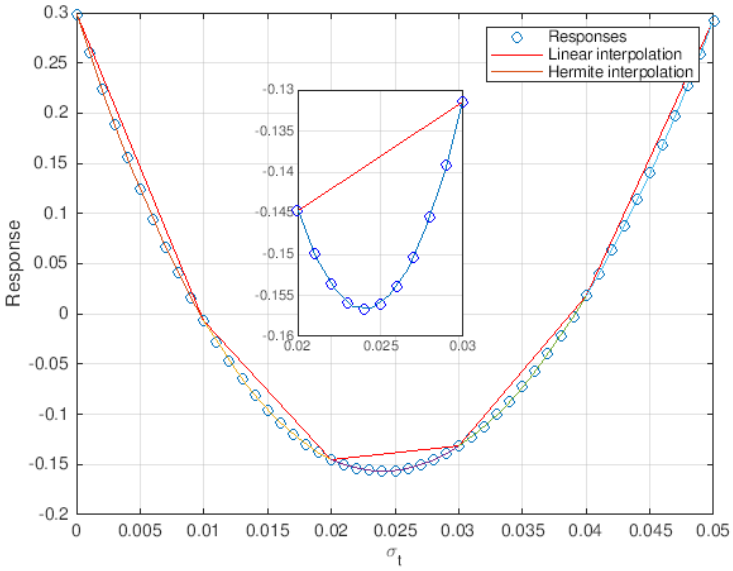


Figure 2.3: An example of a set of responses (circles), a linear interpolation polynomial and a Hermite interpolation polynomial. Interpolation points are $x = [0, 0.01, ..., 0.05]$ The figures shows the linear interpolation simply connecting the interpolation points, whilst Hermite interpolation follows the curvature of the responses. The zoom on $0.02 \leq x \leq 0.03$ especially shows how the linear interpolation is unaware of the curvature.

When more than two function values and derivatives are know, this method can also be used to construct higher order polynomials on the entire domain. In general, if we have m points, we have $2m$ equations and thus can fit a polynomial of order $2m-1$ as we see with two data points (2 data points $\implies$ 4 equations $\implies$ cubic polynomial). Since Hermite interpolation forces the function values and derivatives to be equal in each data point, the found interpolation polynomial will be continuous on the entire domain up to and including first order derivatives.

2.4.2. N-DIMENSIONAL HERMITE INTERPOLATION

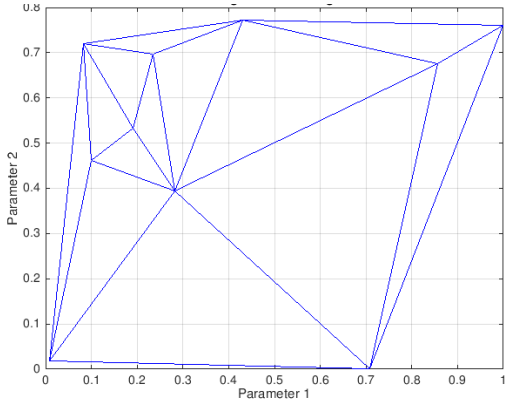
The ideas from Section 2.4.1 can easily be extended to an n-dimensional Hermite interpolation problem. Since there are often multiple patient errors, n-dimensional Hermite interpolation is preferred over one-dimensional Hermite interpolation. For this purpose, an n-dimensional polynomial has to be constructed. In general, an n-dimensional polynomial of order p has $\binom{p+n}{n}$ terms, including the constant term 1. A general n-dimensional polynomial of order p will look accordingly:

$$P_n(x_1, x_2, ..., x_n) = \sum_{p_1=0}^{p} \sum_{p_2=0}^{p-\hat{p}_1} ... \sum_{p_n=0}^{p-\hat{p}_{n-1}} c_{p_1,p_2,...,p_n} (x_1^{p_1} \cdot x_2^{p_2} \cdot ... \cdot x_n^{p_n}) \tag{2.9}$$

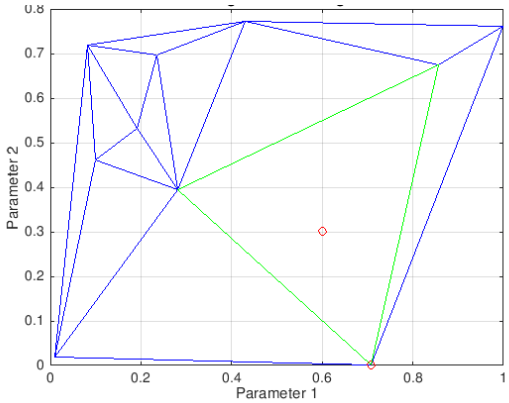
Where $\hat{p}_n = \sum_{i=1}^{n} p_i$ and $c_{p_1,p_2,...,p_n}$ are the coefficients to be determined, i.e. $c_{0,...,0}$ is the constant term. The matrix as in equation 2.8 can be constructed in a similar way. The coefficients are then found by solving the obtained system.

2.4.3. N-DIMENSIONAL SIMPLICES

Every parameter that is subject to an error, contributes to the 'error grid'. If we have m error scenarios, then we will have m coordinates per error parameter. If we have n error parameters, then we have m coordinates in a n dimensional grid. In order to properly fit a polynomial, the obtained grid needs to be divided into subgrids. This will be done by a (Delaunay) triangulation ([12]). A triangulation in higher dimensions is made up of *simplices*. Figure 2.4 shows what this looks like for $m = 10$ and $n = 2$:



(a) Delaunay triangulation of a two dimensional, arbitrary parameter dependant grid with 10 datapoints.



(b) Highlighted datapoint, circle in the middle of triangle, and highlighted triangle in which the datapoint is located. The triangle in which the datapoints is located is easily found using barycentric coordinates, by first finding the closest grid point, highlighted circle-vertex.

Figure 2.4: Representation of a two parameter dependant grid, 2.4a triangulated and 2.4b a highlighted datapoint.

An advantage of a triangulation is that barycentric coordinates ([13]) can be used to determine in which simplex a data point is located. Basically, barycentric coordinates use the vertices of a simplex as basis vectors and write the datapoint (which is located in $(R)^n$ as a linear combination of those basis vectors. If and only if for all weights of the basis vectors λ_i we have $0 \leq \lambda_i \leq 1$, the coordinate is located inside the simplex corresponding to the vertices.

In order to not check for every simplex if a datapoint is inside that simplex, one can first find the closest grid point and only check the simplices adjacent to that grid point. In Figure 2.4b the highlight vertex (Parameter 1 = 0.7, Parameter 2 = 0) only has three adjacent simplices, such that only three simplices need checking instead of thirteen.

In n dimensions, we always need $n+1$ vertices in order to create a simplex to fit a polynomial on: in one dimension (Section 2.4.1) we need 2 vertices, in 2 dimensions we need 3 vertices, etc.. Since in n dimensions we have 1 response value plus n values for the gradient, we obtain $(n+1)\cdot(n+1) = (n+1)^2$ equations (number of vertices times number of sensitivities per point plus the coordinate itself) to fit a polynomial on.

3

METHODS

3.1. DOSE COMPUTATION

The dose is computed using Fortran code written by Lathouwers (Lathouwers [8]). The code solves a discrete version of the Boltzmann transport equation 2.1 using a Galerkin finite element method (Kuzmin [9]). The output of the code is a vector that contains the FEM coefficients, which coincide with the dose at the boundaries of the cell. Since were using a 1D model in this project, a 1D geometry has to be used. The 1D geometry used in this project is given in Figure 3.1:

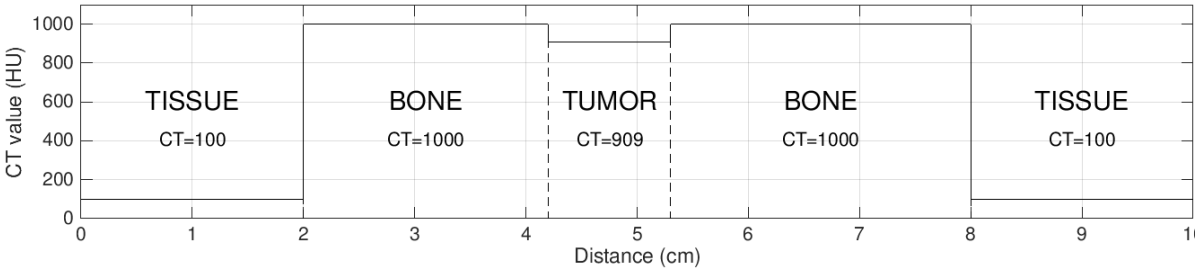


Figure 3.1: Schematic representation of the used geometry. The distribution is as follows: 0 cm to 2 cm: TISSUE, 2 cm to 4.2 cm: BONE, 4.2 to 5.3 cm: TUMOR, 5.3 cm to 8 cm: BONE, 8 to 10 cm: TISSUE.

The reason as to why this geometry is chosen will be explained in section 3.3.2. The vertical axis of figure 3.1 show the CT value in Hounsfield Units [14].

3.1.1. ERROR SCENARIOS

As mentioned in Section 1.2, error scenarios will be simulated. The simulated error scenarios will be in σ_t (same as in Section 2.1.1) ranging from $\sigma_t = 0$ to $\sigma_t = 0.05$. Another error will be in the TUMOR section in Figure 3.1. The CT value will increase from 909 to 999.9 (10% error). This error will simulate tumor shrinkage as the CT value wil come closer to the actual CT value of BONE, implying that the tumor is shrinking. The final error will be in the BONE region between 2 and 4 cm. The CT will decrease from 1000 to 900 (10% error).

3.1.2. FUNCTION FITTING

6 errors will be simulated in σ_t (0 to 0.05 in steps of 0.01), CT_{Tumor} (from 0% error to 10% error in steps of 2%) and CT_{Bone} (from 0% error to 10% error in steps of 2%) so we will have $6^1 = 6$ error scenarios in the 1 dimensional problem, $6^2 = 36$ error scenarios in the 2 dimensional problem and $6^3 = 216$ error scenarios in the 3 dimensional problem. Every error will serve as an additional dimension for the input of our Hermite polynomial $P(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}$. We will fit a function that can relate the error scenario to the dose distribution on those 216 error scenarios. The goodness of the fit will be tested on a set of error scenarios. These errors will then be simulated as follows: $\sigma_t = 0$ to $\sigma_t = 0.05$ in steps of 0.005, CT_{Tumor} will range from 0% error to 10% error in steps of 1% and CT_{Bone} will range from 0% error to 10% error in steps of 1%. We then have $11^3 = 1331$ error scenarios to test the function on.

3.1.3. CREATING A SPREAD OUT BRAGG PEAK

Proton therapy requires a Spread Out Bragg Peak (SOBP). The SOBP can be obtained from the Fortran code that solves the Boltzmann equation. In order to do so, the weights to add the individual pristine peaks have the be determined first. The code is set such that the proton energy width is 2.5 MeV, meaning the 1D-Boltzmann equation is solved for protons with an energy in a range that has a width of 2.5 MeV. From the code, the pristine peaks were obtained for protons with an energy starting at 120 MeV to 80 MeV. Since the width of the energy range is 2.5 MeV, 20 pristine peaks were obtained. Because the pristine peaks are all shifted due to their energy difference, we can add them up using carefully chosen weights and obtain a SOBP (See Paganetti [3]). A MATLAB script (Appendix A) was used in order to find the weights such that the pristine peaks add up to a SOBP.

3.2. REDUCED ORDER MODELLING FOR DOSE DISTRIBUTIONS

The output of the code that solves the discretized Boltzmann equation, the dose distribution vectors, forms the basis for the function fitting. As described in Section 2.2, these dose distributions will be organized in a matrix D. This matrix is then used for the Reduced Order Model (ROM). The actual computation of the ROM is done by MATLAB ([15]).

3.2.1. FINDING THE ACTIVE SUBSPACE

In order to reduce the order of the model, we will find how many singular values, and their corresponding left- and right singular vectors, are actually needed to accomplish a certain precision. Since the singular values decrease in magnitude, we will start with only the largest singular value and add more until the precision conditions are met. The following pseudo-code shows how this process is done (See Appendix B Figure B.1 lines 20 to 30):

Input: Matrices U, S, V with $USV^T = D$, and dose distribution matrix D.
Output: Number of singular values needed to reconstruct dose matrix.
Let i be the number of singular values tested, start with i=1
repeat
  Let $D' = [\mathbf{u}_1 ... \mathbf{u}_i] * [s_1 ... s_i] * ([\mathbf{v}_1 ... \mathbf{v}_i]^T)$,
    with the small bold letters indicating the column vectors of the corresponding matrix.
  Let $\epsilon_{ij} = |(D_{ij} - D'_{ij})/\max(D)|$
    with D_{ij} and D'_{ij} denoting the (i, j)'th element of D and D' respectively.
  Let $\#\epsilon$ = {Number of elements of ϵ such that: $\epsilon_{ij} > 10^{-5}$}
until $\sum \#\epsilon = 0$;

Algorithm 1: Pseudo code that shows how to determine the number of singular values needed for a reconstruction of the dose distribution matrix.

3.3. ADJOINT FLUX CALCULATIONS

Since we want to find the coefficients of the right singular vectors, we shall define the right singular vectors to be our response. Multiplying D by U^T and S^{-1} from the left and then taking the transpose shows $D = USV^T \implies V^T = S^{-1}U^T D \implies V = D^T U(S^{-1})^T = D^T U(S^{-1})$ since S is a diagonal matrix. This means that the (i, j)'th element of V, $V_{i,j}$, can be written as an inner product between the i'th column of D and the product of the j'th mode and the j'th singular value. Using the notation $\mathbf{d}_i$ for the i'th column of D, $\mathbf{u}_j$ for the j'th column of U and s_j for the j'th singular value, we obtain:

$$V_{i,j} = \frac{(\mathbf{d}_i)^T(\mathbf{u}_j)}{s_j} = \frac{<\mathbf{d}_i, \mathbf{u}_j>}{s_j} \tag{3.1}$$

3.3.1. RESPONSE CALCULATIONS

In order to obtain a response and a sensitivity from $V_{i,j} = (\mathbf{d}_i)^T(\mathbf{u}_j)/s_j$ we need to write the inner product as an integral over space, energy and direction. Since the vector $\mathbf{d}_i$ is the discretized dose, according to the FEM we get for the n'th spatial cell:

$$\mathbf{d}_i = \sum_{n=1}^{N} \Big[\int_{x_{n-1}}^{x_n} D_i(x) M_n^{-1} \mathbf{h}_n(x) dx \Big] \mathbf{1}_{(2n-1,2n)} \tag{3.2}$$

Where:

- $\mathbf{1}_{(2n-1,2n)}$ denotes the $(2n-1)$'th and $2n$'th element of the vector $\mathbf{d}_i$.
- M is the mass matrix for the functions $h_{n,2} = \frac{x-x_{n-1}}{x_n-x_{n-1}}$, $h_{n,1}(x) = 1 - h_{n,2}(x)$.
- $D_i(x)$ is the continuous dose *distribution*: $D_i(x) = \iint \big[S(E)\phi(x,E,\hat{\Omega})\big]/dE d\hat{\Omega}$
- x_n and x_{n-1} are the boundaries of spatial cell n.

We let $\mathbf{h}_n = [h_{n,1}(x), h_{n,2}(x)]^T$. Substituting Equation 3.2 into Equation 3.1 yields:

$$\frac{<\mathbf{d}_i, \mathbf{u}_j>}{s_j} = \frac{1}{s_j} \int_0^L D_i(x) \sum_{n=1}^{N} \big[<\mathbf{u}_j, M_n^{-1}\mathbf{h}_n(x)>_n \big] dx \tag{3.3}$$

Where $<\mathbf{u}_j, M_n^{-1}\mathbf{h}_n(x)>_n$ denotes the inner product in spatial cell n. Since $\mathbf{h}_n(x) = 0$ outside spatial cell n, we find that $\sum_{n=1}^{N} \big[<\mathbf{u}_j, M_n^{-1}\mathbf{h}_n(x)>_n \big]$ is a piece wise continuous function and the integral in Equation 3.3 is well defined. If we now plug in the formula for the dose *distribution* into Equation 3.2 we obtain:

$$\frac{<\mathbf{d}_i, \mathbf{u}_j>}{s_j} = \int_0^L \int_{E_{min}}^{E_{max}} \int_{4\pi} \Big[\frac{S(E)}{s_j} \sum_{n=1}^{N} \big[<\mathbf{u}_j, M_n^{-1}\mathbf{h}_n(x)>_n \big] \Big] \phi(x,E,\hat{\Omega}) d\Omega dE dx \tag{3.4}$$

From which we deduce that our *adjoint source* is given by:

$$\Sigma_d = \frac{S(E)}{s_j} \sum_{n=1}^{N} \big[<\mathbf{u}_j, M_n^{-1}\mathbf{h}_n(x)>_n \big] \tag{3.5}$$

In order to find our FEM coefficients for the adjoint source, we multiply with our basis functions again and integrate over a single spatial cell. First we work out the inner product $<\mathbf{u}_j, M_n^{-1}\mathbf{h}_n(x)>_n$. The inverse mass matrix is given by:

$$M_n^{-1} = \frac{1}{(x_n - x_{n-1})} \begin{bmatrix} 4 & -2 \\ -2 & 4 \end{bmatrix} \tag{3.6}$$

From which we can find:

$$\begin{aligned} <\mathbf{u}_j, M_n^{-1}\mathbf{h}_n(x)>_n = & \frac{1}{x_n - x_{n-1}} [\mathbf{U}_{j,2n-1}](4h_1^n(x) - 2h_2^n(x)) \\ & + \frac{1}{x_n - x_{n-1}} [\mathbf{U}_{j,2n}](-2h_1(x) + 4h_2^n(x)) \\ = & \frac{1}{x_n - x_{n-1}} (4[\mathbf{U}_{j,2n-1}] - 2[\mathbf{U}_{j,2n}]) h_1^n(x) \\ & + \frac{1}{x_n - x_{n-1}} (-2[\mathbf{U}_{j,2n-1}] + 4[\mathbf{U}_{j,2n}]) h_2^n(x) \end{aligned} \tag{3.7}$$

Where $\mathbf{U}_{j,k}$ denotes the k'th element of the j'th column vector of $\mathbf{U}$. We now multiply Equation 3.7 by $\mathbf{h}_n(x)$ and integrate over $[x_{n-1}, x_n]$, such that for every cell we get as our FEM coefficients:

$$\int_{x_{n-1}}^{x_n} \frac{S(E)}{s_j} \sum_{n=1}^{N} \big[<\mathbf{u}_j, M_n^{-1}\mathbf{h}_n(x)>_n \big] \mathbf{h}_n(x) dx = \frac{1}{s_j} \begin{bmatrix} \mathbf{u}_{j,2n-1} \\ \mathbf{u}_{j,2n} \end{bmatrix} \tag{3.8}$$

3.3.2. SENSITIVITY CALCULATIONS

As described in Section 2.3.2 the sensitivities can be found by equation 2.7. In this project three sensitivities are calculated, two of which are essentially identical. One with respect to σ_t and two with respect to the perturbation in the CT scan. As described in Section 2.3.2 we need to find the derivative of the operator with respect to σ_t and the perturbation in the CT values. In this project the operator is given by the Boltzmann operator:

$$B = -\mu\frac{\partial}{\partial x} + \sigma_t + S(E)\frac{\partial}{\partial E} \tag{3.9}$$

Such that we need to solve the following equation (Kasbergen [2]):

$$B\phi = S \tag{3.10}$$

With S the adjoint source as in Equation 3.5. Equation 3.10 explains the name adjoint source for S as S is seen as a source for protons.

SENSITIVITY WITH RESPECT TO σ_t

As in Equation 2.7 we need to find the partial derivative of the operator with respect to a parameter. It turns out that for the sensitivity with respect to σ_t, this partial derivative is equal to 1 (partial derivative of Equation 3.9 with respect to σ_t) such that:

$$\frac{\partial V_{ij}}{\partial \sigma_t} = -<\phi_0^\dagger, \phi_0> \tag{3.11}$$

The perturbation in σ_t will run from $\sigma_t = 0$ (nominal scenario) to $\sigma_t = 0.05$.

SENSITIVITY WITH RESPECT TO STOPPING POWER

The sensitivity with respect to the perturbation in the CT scan is a little more work to compute. The stopping power is the product of the density with the (chemical) composition $S_n(E) = \rho_n \cdot C$ with ρ_n the density per spatial cell. According to Schneider (Schneider *et al.* [16]), the density can be found from CT-values and the chemical composition per spatial cell is made up of the weights given by source [16, p. 475, table 6] that in turn also depent on the CT-value. The range in Hounsfield Units for which the chemical composition vector is constant is in the order of 10^2 Hounsfield units. The range in which the density is constant is considerably smaller, only about one Hounsfield Unit, we shall assume the following:

A perturbation in the CT-values shall only perturb the density.

Since $S_n(E) = \rho_n \cdot C$ we find $\Delta S(E) = \Delta\rho \cdot C$. Since dose is defined as $Dose = \iiint \left[S(E)\phi(x,E,\hat{\Omega})\right]/\rho dx dE d\hat{\Omega}$. We see that the perturbation in ρ cancels and the total dose remains unchanged! The dose distribution is only shifted left or right, that is, the plateau in the SOBP is positioned deeper or shallower in the patient's body.

Since we have found that our operator is given by Equation 3.9, we need to find the partial derivative of B with respect to CT-value perturbation which will only be in $S(E)$. This will be done by a simple finite difference scheme: $\frac{\partial S(E)}{\partial CT} = \frac{S_p(E) - S_0(E)}{CT}$ where CT is the percentual perturbation in the CT values and $S_p(E)$ the stopping power corresponding to the error scenario, such that:

$$\frac{\partial V_{ij}}{\partial CT} = -<\phi_0^\dagger, \left[\frac{S_p(E) - S_0(E)}{CT}\right]\phi_0> \tag{3.12}$$

Two regions of CT-values will be perturbed. One region will be the entire tumor region (CT_{Tumor}), simulating tumor shrinkage, whilst the other region will be the entire region in front of the tumor between 2cm and 4.2 cm CT_{Tissue}. A schematic representation of the geometry used is given in Figure 3.1.

The CT-value of 909 for the Tumor region is chosen such that a 10% error in that value stays just under a CT-value of 1000, which is needed to ensure a perturbation in CT-value only perturbs the density.

3.4. HERMITE INTERPOLATION

In this project we used a full tensorised grid. The obtained grid is triangulated by MATLAB. On every simplex (Section 2.4) (note that a line segment in 1D is also a simplex) a Hermite polynomial is fitted using the method described in Section 2.4. If the found system turns out the be singular, or the matrix computation turns out to be close to singular, then the entire system can be solved by an SVD decomposition on the interpolation matrix. The interpolation matrix can for instance be singular when the columns or rows are dependant. After some thorough inspection, by M. van Gijzen, of the interpolation matrix on a two dimensional simplex using the polynomial $P_2(x_1, x_2) = \sum_{i=0}^{2}\sum_{j=0}^{2} c_{ij} x_1^i x_2^j$, it was concluded that the two dimensional interpolation matrix is dependant with the given polynomial. To circumvent this, we chose our polynomial to be $\hat{P}_2(x_1, x_2) = P_2(x_1, x_2) - c_{22}x_1^2x_2^2 + c_{22}x_1^3x_2^3$. In other words, we replace the term $c_{22}x_1^2x_2^2$ with $c_{22}x_1^3x_2^3$ The full code used for the computation of the polynomial coefficients can be found in Appendix B.

In order to validate the found Hermite polynomial, 11 error scenarios in each dimensional will be simulated ($11^1 = 11$ in one dimension, $11^2 = 121$ in two dimensions and $11^3 = 1331$ in three dimensions). The new reconstructed dose distribution matrix will be compared with the simulated dose distribution matrix. From the simulated dose distribution matrix, the coefficient matrix V can also be obtained by the formula: $V = D^T U(S^{-1})$ such that calculated and simulated V matrices can also be compared. Essentially, V is what we want to determine since the dose distribution matrix follows form this.

However, there is a subtle issue that needs addressing. In one dimension, a simplex has two vertices and so we have four data points (2 function values and 2 derivatives), on which we can unambiguously fit a third order (Hermite) polynomial. In two dimensions, a simplex has 3 vertices and so we have nine data points (3 function values, 3 derivatives in both dimensions), such that we can fit a polynomial that is the product of two second order polynomials, i.e. $P_2(x_1, x_2) = (a_0 + a_1x_1 + a_2x_1^2)(b_0 + b_1x_2 + b_2x_2^2)$ (NOTE: $P_2(x_1, x_2)$ as given here is not a general 2-dimensional polynomial of order 4 since there are no terms $x_1^3x_2$, $x_1x_2^3$, x_1^4 and x_2^4). In general we have, in dimension d, $(1+d)^2$ equations to fit a polynomial on. We can only fit a polynomial as a product of one dimensional, polynomials as done before, if $\sqrt[d]{(1+d)^2} \in N$. That is, if $\sqrt[d]{(1+d)^2}$ is an integer. We see for $d > 2$, that $\sqrt[d]{(1+d)^2}$ is not an integer and so we cannot use this method unambiguously. Since in this project we do not go higher than third order, we solve this issue by ignoring all third order terms (x^3, y^3, z^3 and xyz). However, for higher orders the difference between number of equations and number of terms in the polynomial will grow, resulting in an unreliable solution.

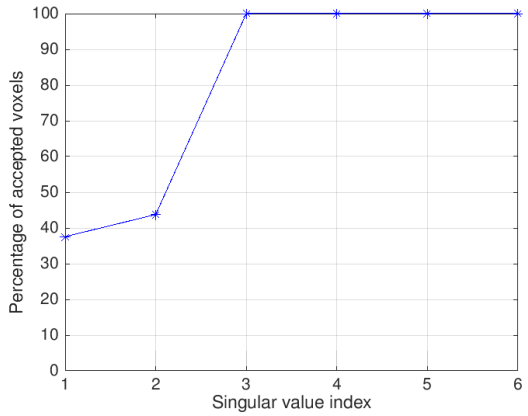
4

RESULTS AND DISCUSSION

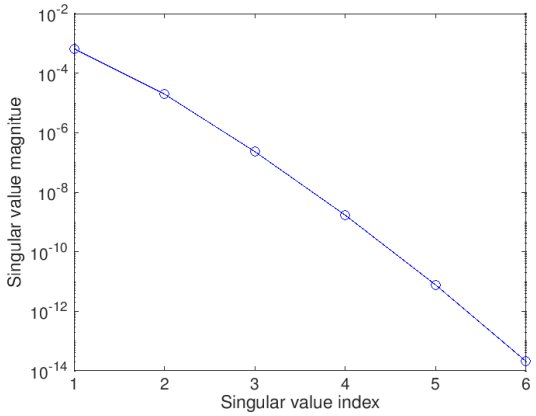
In this section we will present the results of the computation of a Hermite polynomial, the reconstruction of the dose distribution matrix and the reconstruction of the coefficients matrix V. The elements of the matrix V are also called the responses (from Section 2.3.1).

4.1. DOSE RECONSTRUCTION WITH ONE ERROR

In one dimension, 3 singular values were needed to reconstruct the entire dose distribution matrix D to within 1% precision. (See Figure 4.1). This means 3 columns of V had to be reconstructed. Note that the length of the columns of V now indicates how many error scenarios we reconstruct, such that the length will be 11.



(a) Percentage of accepted voxels versus the singular value index. 3 singular values, and thus modes, are needed to reconstruct the dose distribution matrix to within 1% accuracy.



(b) Magnitude of the singular values versus the singular value index.

Figure 4.1: Results for the ROM on doses with error in one dimension. Only 3 modes are needed to reconstruct full dose distribution matrix to within 1% accuracy.

4.1.1. CONSTRUCTION OF COEFFICIENT MATRIX V

Figure 4.2 shows the reconstruction of the first order coefficients of V. We see that the construction of the first column of the V matrix, with $R^2 = 1$, went pretty well. The error is about 8 orders of magnitude smaller than the actual values. We notice that the error seems random. There is no clear pattern emerging in the error. Note that the difference between the maximum value and the minimum value of the coefficients is only about 0.08, from which we conclude (especially in contrast with Figures 4.3 and 4.4) that the first order coefficients are, more or less, constant with respect to the other modes.

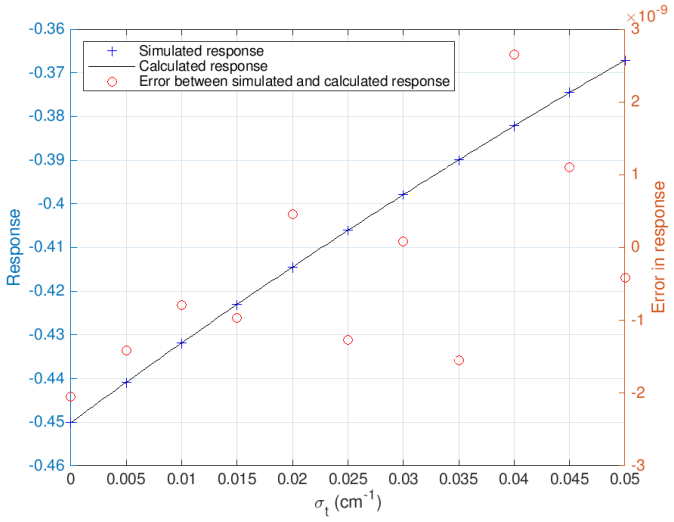


Figure 4.2: Reconstruction of the first order coefficients. Solid line shows the simulated results (left axis), '+' marks shows the calculated results (left axis) ans the 'o' marks show the errors (right axis). $R^2 = 1$.

The following figure shows the reconstruction of the second order coefficients:

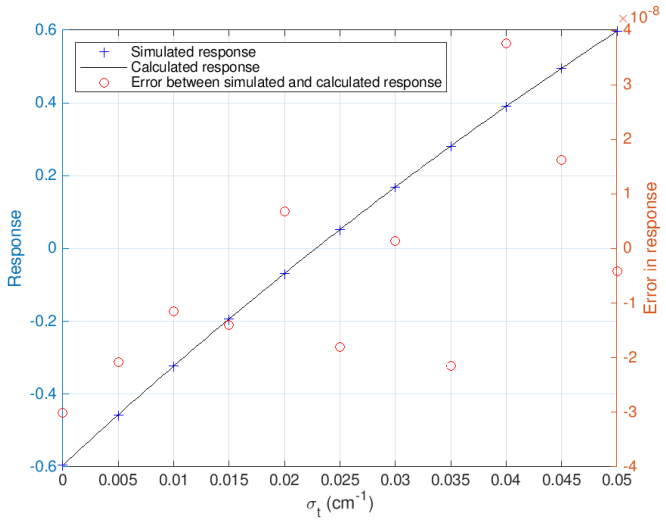


Figure 4.3: Reconstruction of the second order coefficients. $R^2 = 1$.

Again we see an R-square value of $R^2 = 1$. Comparing the order of magnitude of the error with the coefficient values we notice a difference of about 7 orders of magnitude. We can again conclude that the error is random and there is no clear pattern emerging in the error.

Figure 4.4 shows the results for the third order coefficients. For the third time, we have $R^2 = 1$. A rather amazing result. All three sets of coefficient can be approximated with an R^2 of 1. The order of magnitude of the error is again about 7 orders smaller than the coefficient values. We do notice that the coefficients of mode 3 have a lot more curvature to them then the coefficients of other modes. However, the sensitivities from the adjoint analysis in combination with the Hermite polynomial can approximate the coefficients really well. Notice that the errors in the third mode also do not show a pattern, the average seems to be around 0 (notice a shifted right axis ranging from -10 to 6 in Figure 4.3).

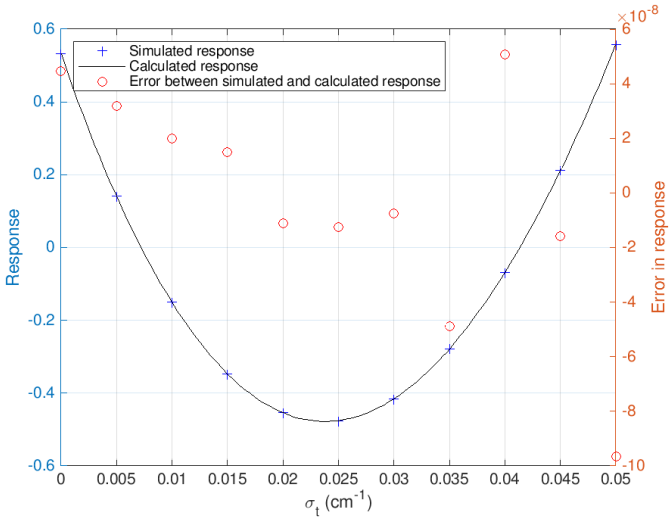


Figure 4.4: Reconstruction of the third order coefficients. $R^2 = 1$.

4.1.2. CONSTRUCTION OF THE DOSE DISTRIBUTION MATRIX

We will now focus on the reconstruction of the dose distribution matrix D. Recall from Section 2.2, that $D = USV^T$. Since U and S contain the modes in 1 dimension, it is the coefficient matrix V that determines the results. We shall start by discussing the nominal scenario shown in Figure 4.5:

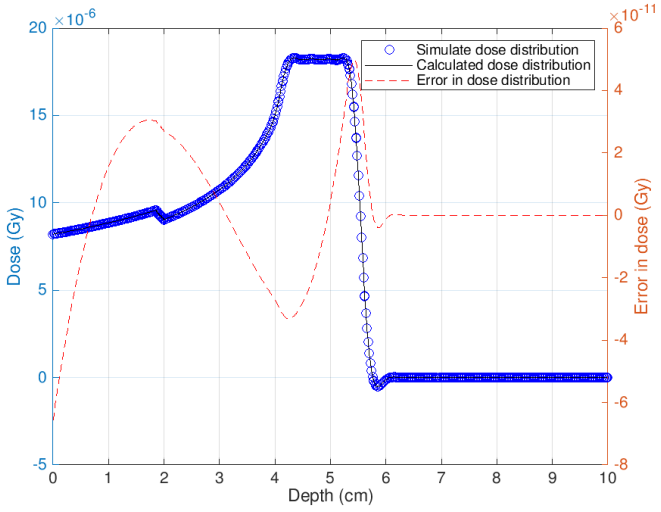
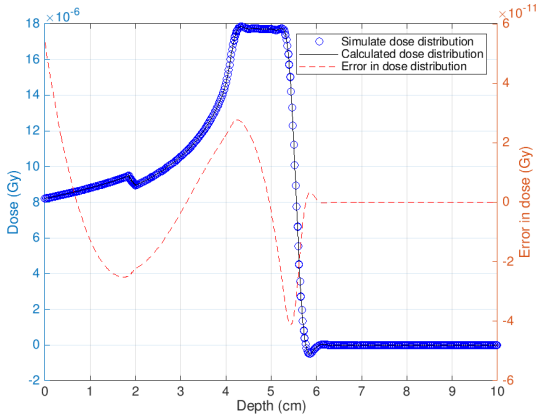
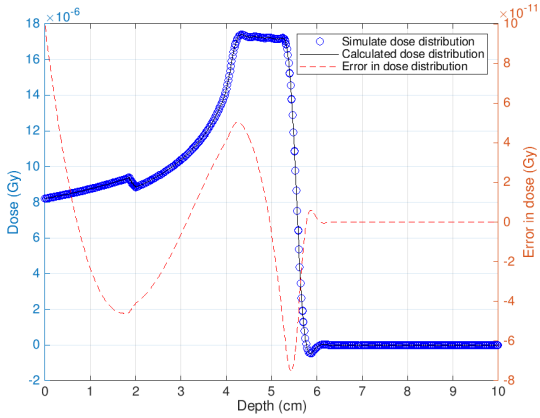


Figure 4.5: Reconstruction of the nominal scenario in 1 dimension. Figure contains 3 plots; a plot of the simulated dose distribution, a plot of the reconstructed dose distribution (actually right on top of the first graph) and the error between them. $R^2 = 1$.

First we need to comment on the shape of the graph. Around 2 cm we see a little drop in the calculated and simulated dose distributions. This is due to the CT values dropping of there. The second drop in the graph is just before 6 cm where the dose distributions become negative! This is a results of the FEM and it should be clear that it is unphysical to have a negative dose. We see that the error in the nominal scenario ($\sigma_t = 0$) is 5 orders of magnitude smaller than the actual dose distribution. This results, yet again, in $R^2 = 1$. It should be noted that the nominal scenario goes to 0 after 8cm, a result of proton therapy, which greatly improves the R^2 value. Figures 4.6 show the dose reconstructions for the remaining 10 scenarios.

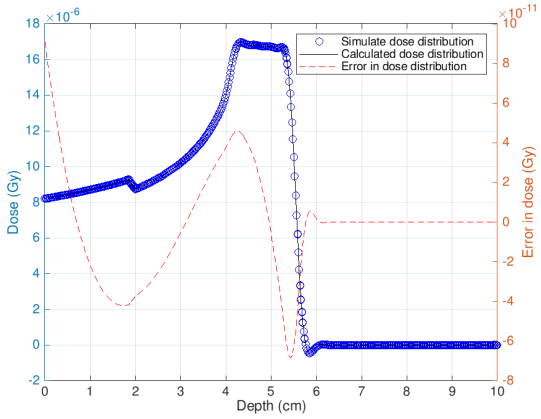


(a) Dose reconstruction for $\sigma_t = 0.005$. $R^2 = 1$.

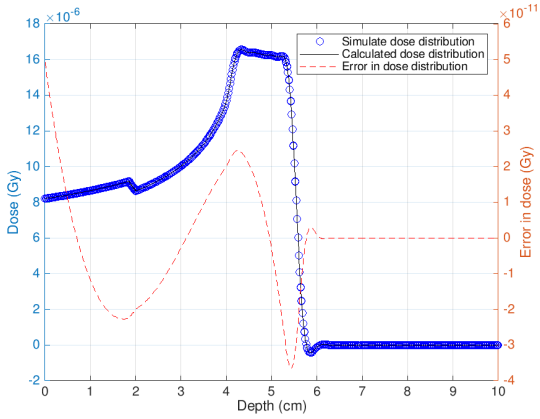


(b) Dose reconstruction for $\sigma_t = 0.01$. $R^2 = 1$.

Figure 4.6: Results of the construction of the second and third scenario in 1 dimension.

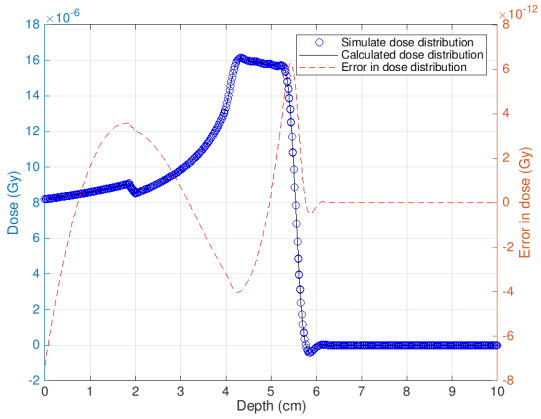


(c) Dose reconstruction for $\sigma_t = 0.015$. $R^2 = 1$.

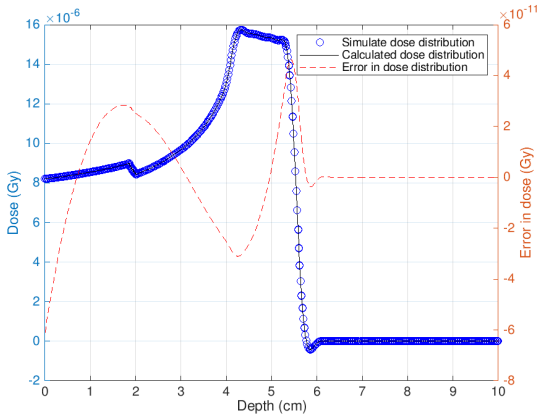


(d) Dose reconstruction for $\sigma_t = 0.02$. $R^2 = 1$.

Figure 4.6: Results of the construction of the fourth and fifth error scenario in 1 dimension.

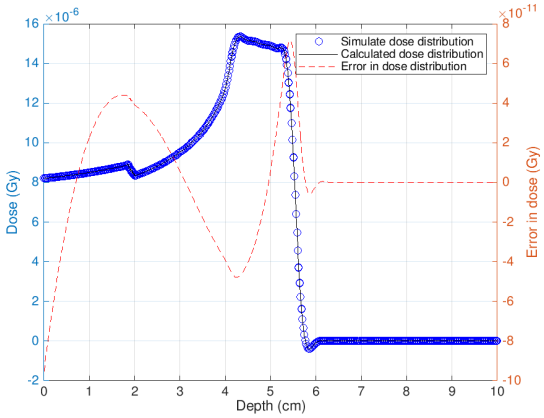


(e) Dose reconstruction for $\sigma_t = 0.025$. $R^2 = 1$.

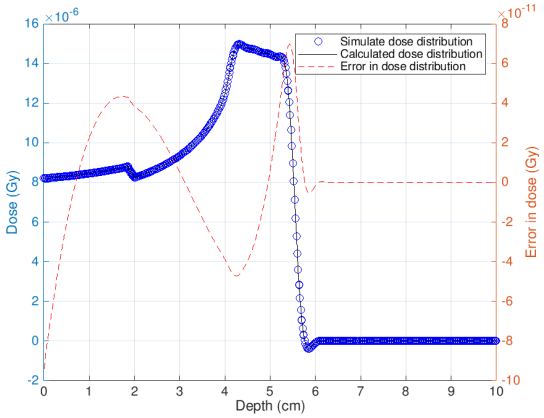


(f) Dose reconstruction for $\sigma_t = 0.03$. $R^2 = 1$.

Figure 4.6: Results of the construction of the sixth and seventh error scenario in 1 dimension.

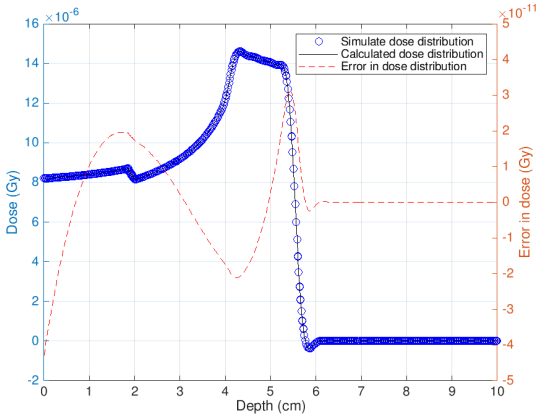


(g) Dose reconstruction for $\sigma_t = 0.035$. $R^2 = 1$.

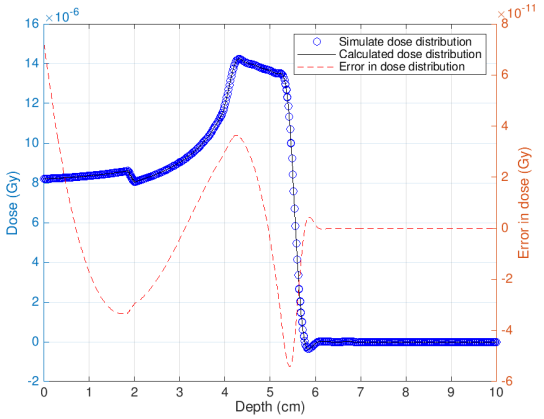


(h) Dose reconstruction for $\sigma_t = 0.04$. $R^2 = 1$.

Figure 4.6: Results of the construction of the eighth an ninth error scenario in 1 dimension.



(i) Dose reconstruction for $\sigma_t = 0.045$. $R^2 = 1$.



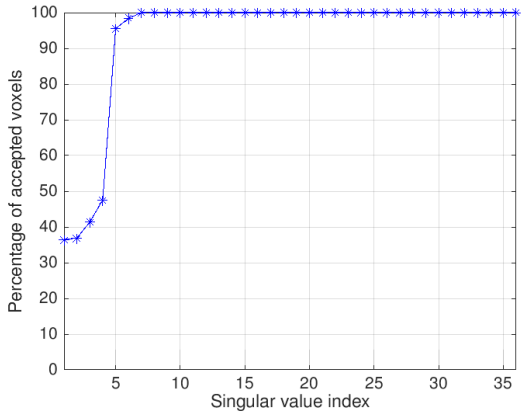
(j) Dose reconstruction for $\sigma_t = 0.05$. $R^2 = 1$.

Figure 4.6: Results of the construction of the tenth and eleventh error scenario in 1 dimension.

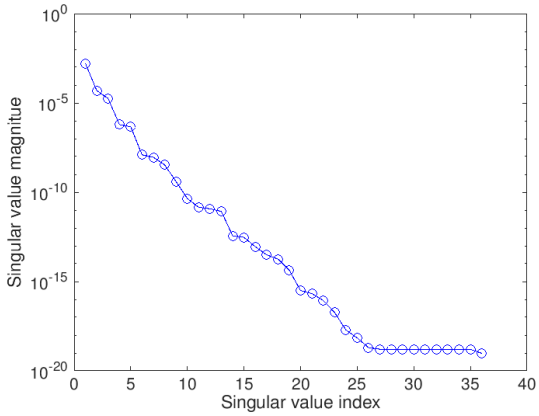
We notice, once again, that for all dose reconstructions we have $R^2 = 1$. We do notice that the errors look quite similar. This is because the error is a linear combination of the modes that are not used in the reduced order model (in this case modes 4 and higher), so the errors consist of linear combination of the same shape and thus look alike.

4.2. DOSE RECONSTRUCTION WITH TWO ERRORS

In two dimensions, 7 modes are needed to reconstruct the dose distribution matrix as shown in Figure 4.7.



(a) Percentage of accepted voxels versus the singular value index. 7 singular values are needed to reconstruct the dose distribution matrix to within 1% accuracy.

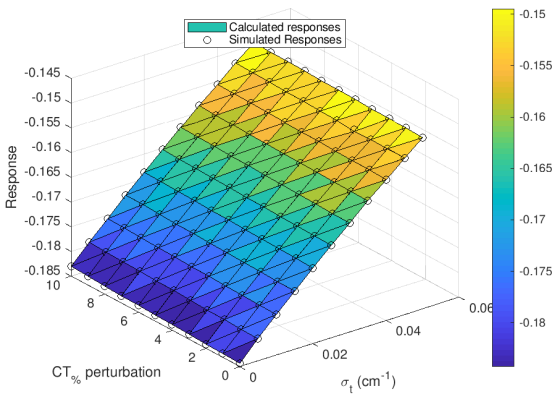


(b) Magnitude of the singular values versus the singular value index.

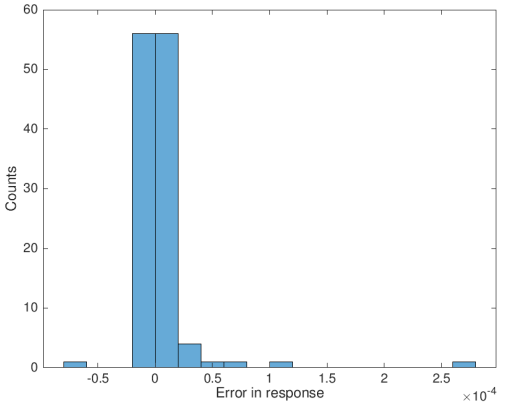
Figure 4.7: Results for the ROM on dose distribution matrix with errors in two dimensions. 7 modes are needed to reconstruct the dose distribution matrix to a precision of 1%.

4.2.1. CONSTRUCTION OF COEFFICIENT MATRIX V

We shall start by looking at the reconstruction of the first mode coefficients:



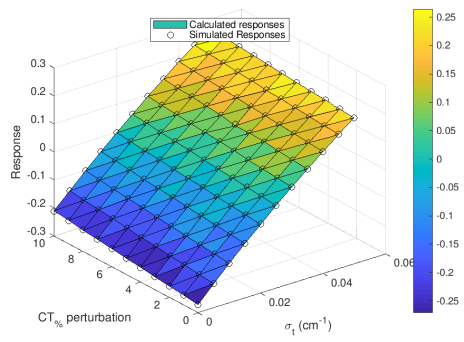
(a) Reconstruction of the first order coefficients. The plane marks the constructed coefficients, while the 'o' marks show the simulated coefficients. $R^2 = 1$.



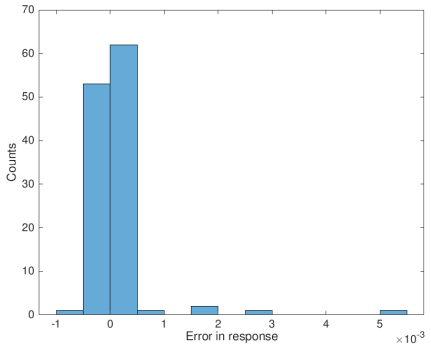
(b) Histogram of the errors in the reconstruction of the first order modes. Note that the error is in the order of magnitude 10^{-4}, about 3 orders lower that the coefficients.

Figure 4.8: Results of the construction of the first order mode coefficients in 2 dimensions. Note that the triangulation is clearly visible.

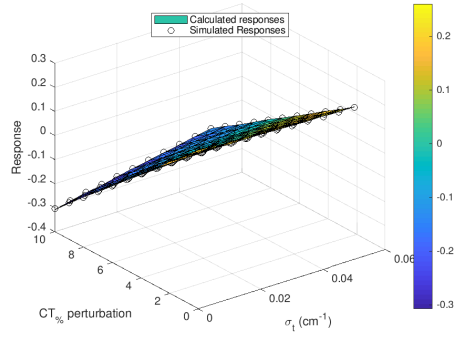
We note that we have $R^2 = 1$, and that the difference between the maximum and minimum value for the coefficients is about 0.04. We again see, compared to the other modes, that the first mode is more or less constant. We note too, that the error is in the order of magnitude 10^{-5}, whilst the coefficients are in the order of magnitude 10^{-1}. We shall now present the results for the higher order modes.



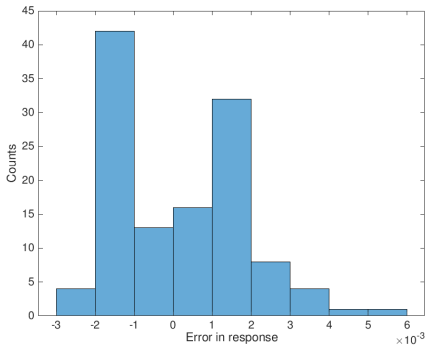
(c) Reconstruction of the second order coefficients. The plane marks the constructed coefficients, while the 'o' marks show the simulated coefficients. $R^2 = 1$.



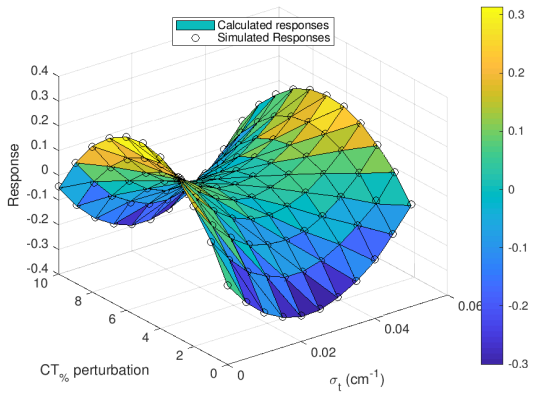
(d) Histogram of the errors in the reconstruction of the second order modes. Note that the error is at most in the order of magnitude 10^{-3}, only 2 orders lower that the coefficients.



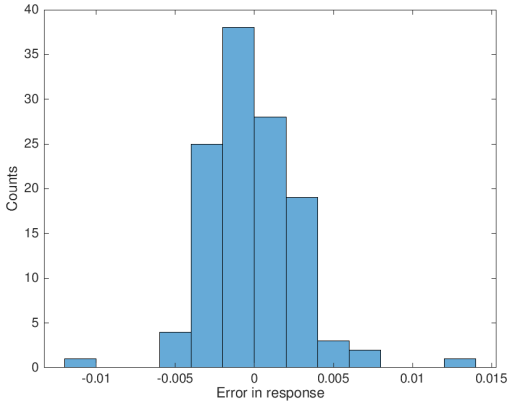
(e) Reconstruction of the third order coefficients. The plane marks the constructed coefficients, while the 'o' marks show the simulated coefficients. $R^2 = 1$.



(f) Histogram of the errors in the reconstruction of the third order modes. The error is order of magnitude 10^{-3}, 2 orders lower than the coefficients.

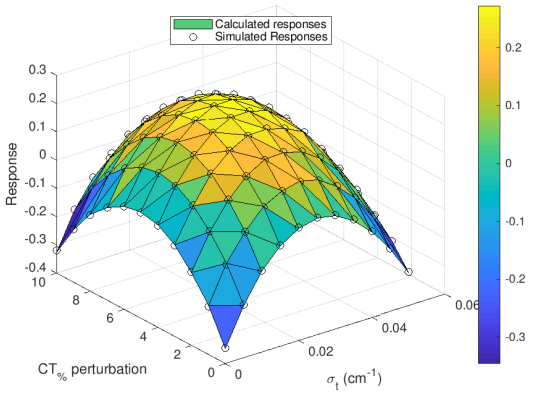


(g) Reconstruction of the fourth order coefficients. The plane marks the constructed coefficients, while the 'o' marks show the simulated coefficients. $R^2 = 0.999$.

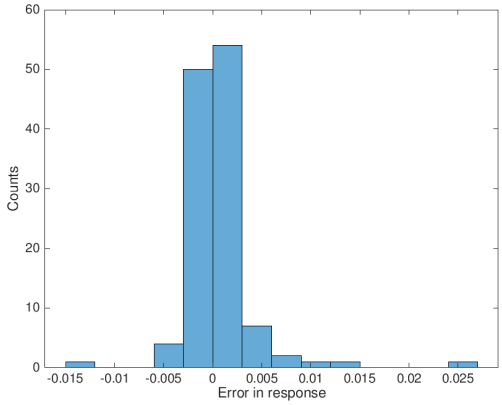


(h) Histogram of the errors in the reconstruction of the fourth order modes. The error seems concentrated around 0. It can however reach values of -0.01 or 0.015 which is a relative error of about 100%.

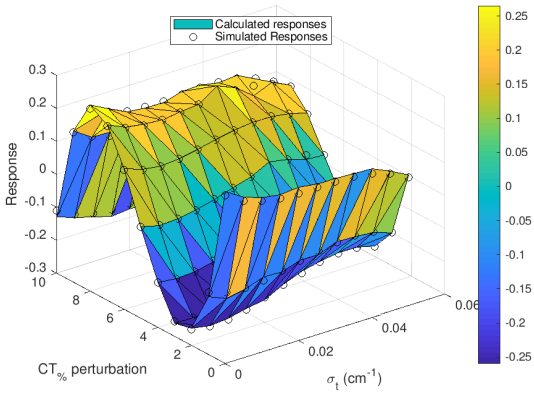
Figure 4.8: Results of the construction of the second, third and fourth order mode coefficients in 2 dimensions.



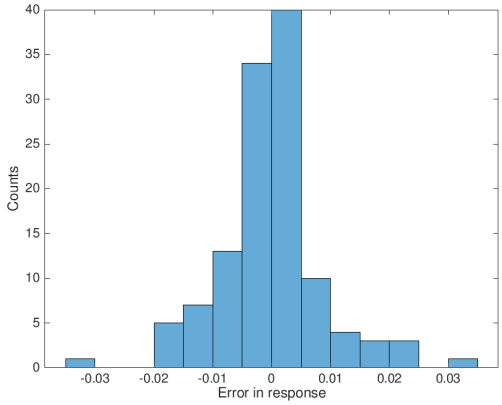
(i) Reconstruction of the fifth order coefficients. The plane marks the constructed coefficients, while the 'o' marks show the simulated coefficients. $R^2 = 0.999$.



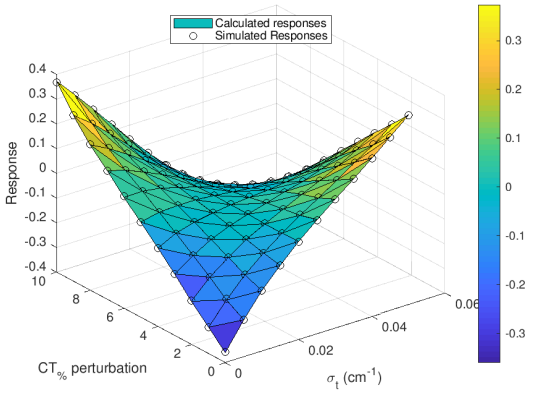
(j) Histogram of the errors in the reconstruction of the fourth order modes. The error seems concentrated around 0 except for 1 case where it is 0.08.



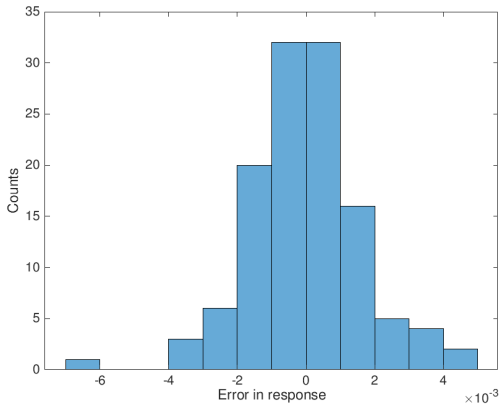
(k) Reconstruction of the sixth order coefficients. The plane marks the constructed coefficients, while the 'o' marks show the simulated coefficients. $R^2 = 0.996$



(l) Histogram of the errors in the reconstruction of the sixth order modes. The errors in this mode are by far the largest, only one order of magnitude smaller than the coefficients.



(m) Reconstruction of the seventh order coefficients. The plane marks the constructed coefficients, while the 'o' marks show the simulated coefficients. $R^2 = 1$



(n) Histogram of the errors in the reconstruction of the seventh order modes. The error is in the order of magnitude 10^{-3}, again 2 orders less than the coefficients.

Figure 4.8: Results of the construction of the fifth, sixth and seventh order mode coefficients in 2 dimensions.

Overall we note that R^2 is about 1 for all modes, except for the sixth where it is 'only' 0.996. This relatively low value for R^2 for the sixth mode can be explained by the oscillatory nature of the coefficients. Since we are fitting polynomials, which are not particularly oscillatory, we can explain this value. What is sort of surprising, is that for the seventh mode we have $R^2 = 1$ again. We would expect higher modes to have more complicated shapes in order to conserve the orthonormality discussed in Section 2.2.2. The same holds for modes 4 and 5, the shapes of the coefficients has an oscillatory nature which makes it hard to fit polynomials on. In the end, the results look promising in the reconstruction of the dose distribution matrix, which we shall discuss next.

4.2.2. CONSTRUCTION OF THE DOSE DISTRIBUTION MATRIX

Since we have 11 error scenarios in each dimension, we have 121 error scenarios in total. For brevity, we shall not show them all here but stick to some special cases. The special cases being the nominal scenario, the two worst approximated scenarios and the scenario with the highest simulated error, that is $\sigma_t = 0.05$ and $CT_{Tumor} = 10\%$. We shall start with the nominal scenario:

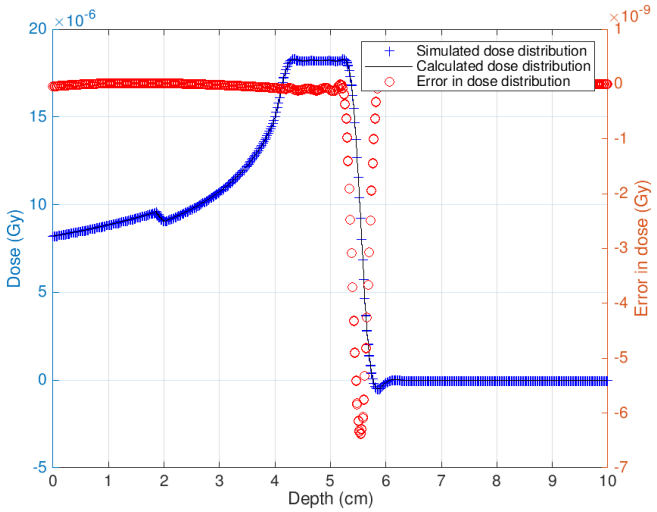
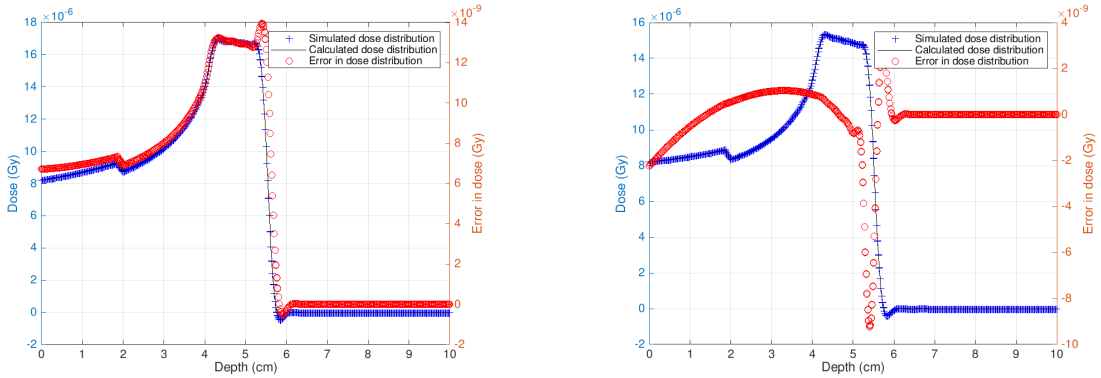


Figure 4.9: Dose reconstruction of the nominal scenario. The error shows a drop when the dose drops to zero, indicating a slight difference in the simulate dose and the constructed dose for the drop off area. $R^2 = 1$.

Again, we see that the reconstruction of the nominal scenario went pretty well. The error has a peak value with an order of magnitude that is three orders lower than the actual dose distribution. The drop in the error is a phenomena that occurs often in the dose reconstruction for an error in two dimensions. For brevity, we have not shown all results here. We shall now analyze the two worst approximated scenarios:



(a) Error scenario $\sigma_t = 0.005$, $CT_{Tumor} = 3\%$. $R^2 = 1$

(b) Error scenario $\sigma_t = 0.025$, $CT_{Tumor} = 7\%$. $R^2 = 1$

Figure 4.10: Worst approximated dose distributions. Scenarios $\sigma_t = 0.005$, $CT_{Tumor} = 3\%$ and $\sigma_t = 0.025$, $CT_{Tumor} = 7\%$. Both still have $R^2 = 1$.

To start with, we notice that both R^2 values are 1. Since these are the worst case scenarios, that means all the other scenarios are better. But better than 1 can not be, so all scenarios have $R^2 = 1$. The actual R^2 values are 0.9999995 (for $\sigma_t = 0.005$, $CT_{Tumor} = 3\%$) and 0.999991 (for $\sigma_t = 0.025$, $CT_{Tumor} = 7\%$). For the first time, we notice a systematic error in Figure 4.10a. We notice that the error has the shape of a SOBP, and that it is mostly located above the simulated dose distribution. The latter implies that we the approximation is predicting a lower dose than we are actually measuring. We shall now discuss the scenario with the highest error in Figure 4.11:

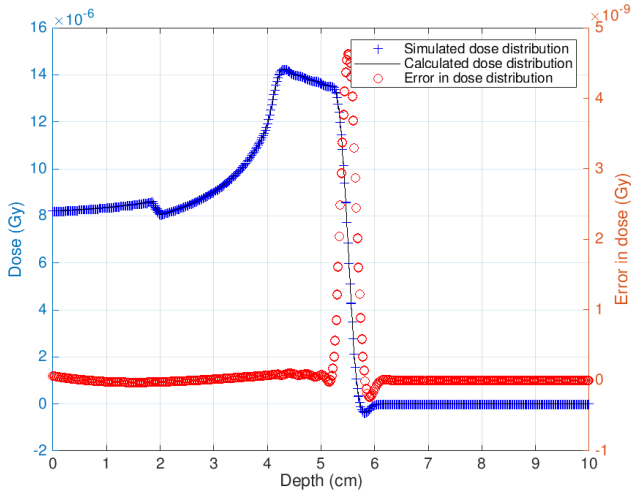
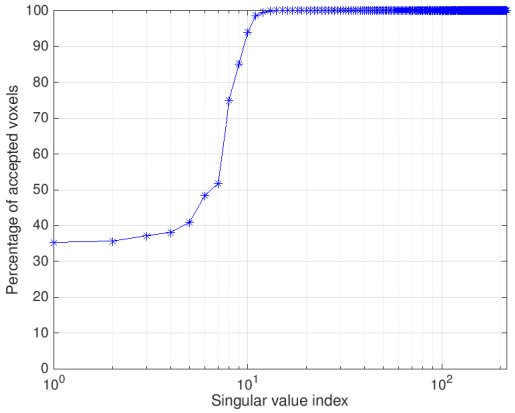


Figure 4.11: Error scenario $\sigma_t = 0.05$, $CT_{Tumor} = 10\%$. We again see a peak when the dose drops. $R^2 = 1$.

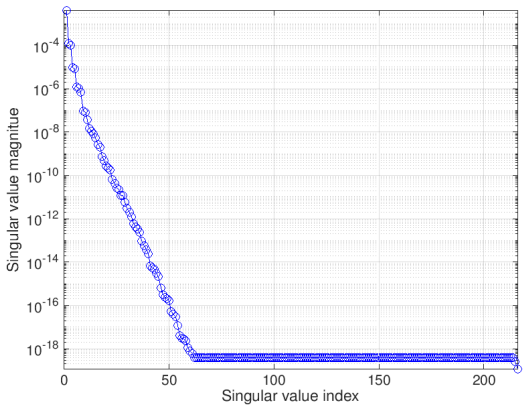
The error in the constructed dose distribution has a peak when the dose drops off. However, we still see $R^2 = 1$ even for the scenario with maximum error. We thus see we are able to approximate every dose distribution to within acceptable margin with an error in two dimensions.

4.3. DOSE RECONSTRUCTION WITH THREE ERRORS

For errors in three dimensions we need 14 modes to reconstruct the dose distribution matrix as seen in Figure 4.12.



(a) Percentage of accepted voxels versus the singular value index. 14 singular values are needed to reconstruct the dose distribution matrix to within 1% accuracy. Note the logarithmic singular value index axis.



(b) Magnitude of the singular values versus the singular value index. The magnitude seems to decrease exponentially. This decay stops when machine precision is reached. This is what explains the constant section of the graph.

Figure 4.12: Results for the ROM on dose distribution matrix with errors in two dimensions. 14 modes are needed to reconstruct the dose distribution matrix to a precision of 1%. The x-axis in figure 4.12a has been plotted on a logarithmic scale to visualize the acceptance rate.

4.3.1. CONSTRUCTION OF COEFFICIENT MATRIX V

We shall start by examining the construction of the coefficient matrix. Since we are considering 4-dimensional data (Errors in three dimensions ans the responses) there is no other way, other than a 3 dimensional scatter plot with color coded values, than to show a histogram of the error, the R^2 values and draw conclusions from there. We shall start by examining the first mode coefficients:

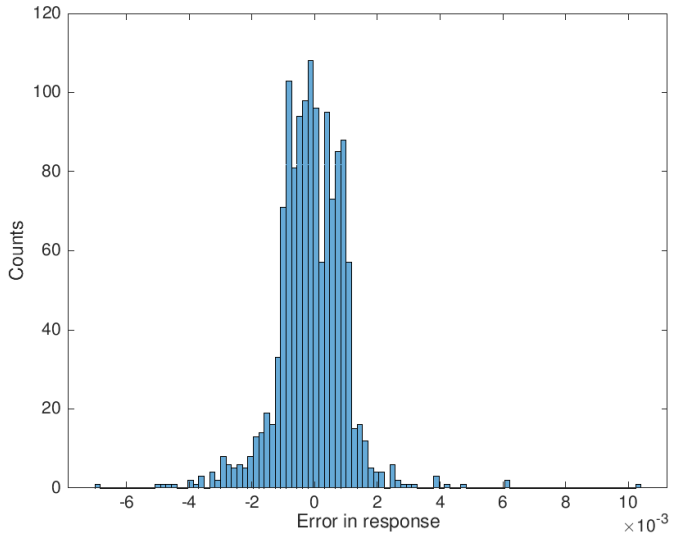
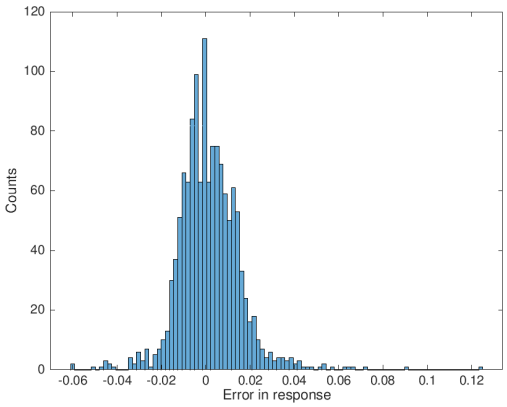
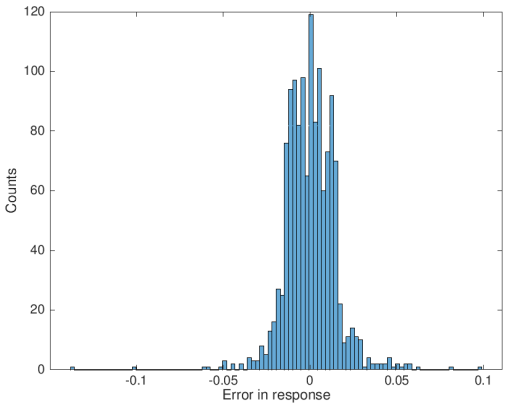


Figure 4.13: Histogram of the error in the construction of the first mode coefficients. We have $R^2 = 0.939$.

We see that $R^2 = 0.939$, which makes sense considering the histogram centers around 0 and has a maximum error of 0.01. Comparing this value with earlier values for R^2 found we note it is a little low. This can be explained by the problem described in Section 2.4. We are losing information due to the SVD in the Hermite interpolation vector product equation from the Hermite polynomial fitting. We shall now present the other 13 modes and comment on them:

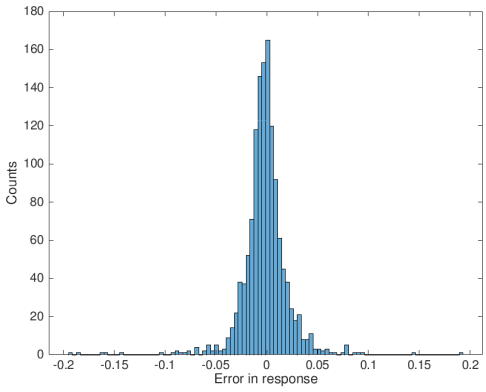


(a) Histogram of the error in the construction of the second mode coefficients. We have $R^2 = 0.948$.

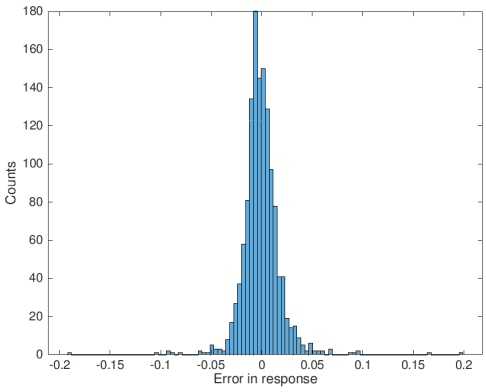


(b) Histogram of the error in the construction of the third mode coefficients. We have $R^2 = 0.945$.

Figure 4.13: Histograms of the error in the construction of the second and third mode.

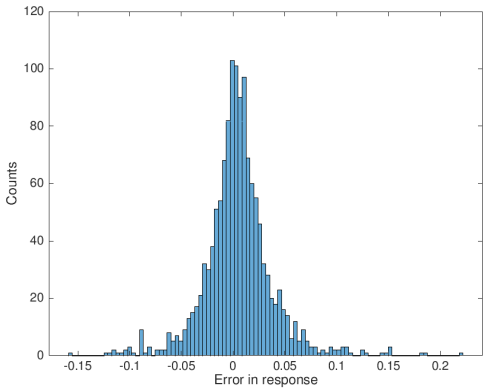


(c) Histogram of the error in the construction of the fourth mode coefficients. We have $R^2 = 0.852$.

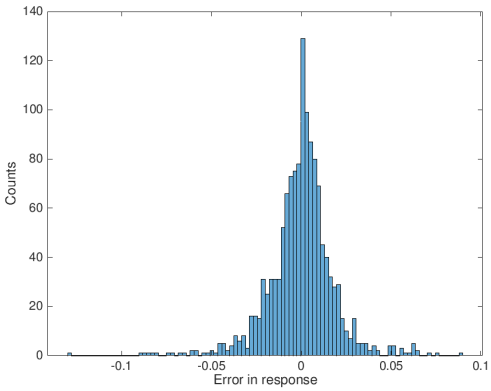


(d) Histogram of the error in the construction of the fifth mode coefficients. We have $R^2 = 0.900$.

Figure 4.13: Histograms of the error in the construction of the fourth and fifth mode.

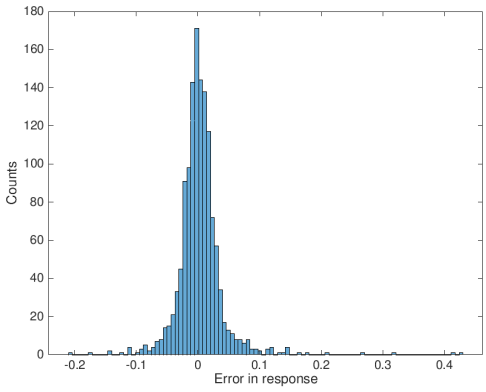


(e) Histogram of the error in the construction of the sixth mode coefficients. We have $R^2 = 0.740$.

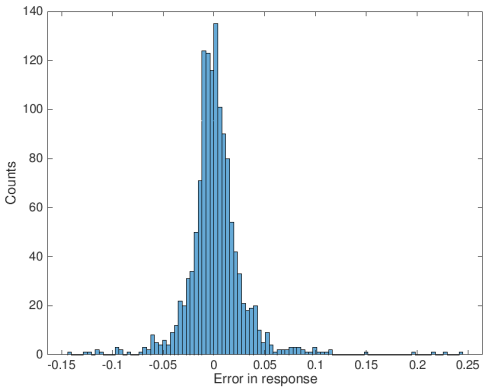


(f) Histogram of the error in the construction of the seventh mode coefficients. We have $R^2 = 0.901$.

Figure 4.13: Histograms of the error in the construction of the sixth and seventh mode.

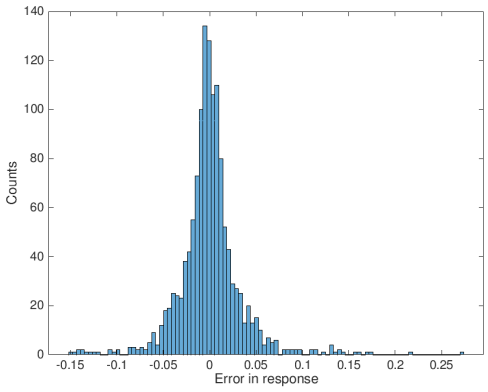


(g) Histogram of the error in the construction of the eighth mode coefficients. We have $R^2 = 0.569$.

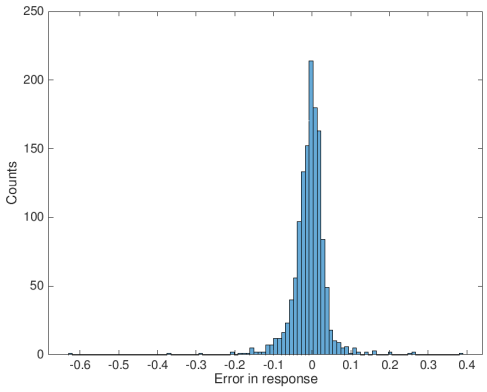


(h) Histogram of the error in the construction of the ninth mode coefficients. We have $R^2 = 0.781$.

Figure 4.13: Histograms of the error in the construction of the eighth and ninth mode.

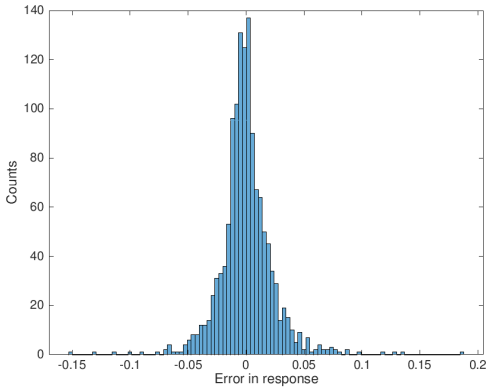


(i) Histogram of the error in the construction of the tenth mode coefficients. We have $R^2 = 0.694$.

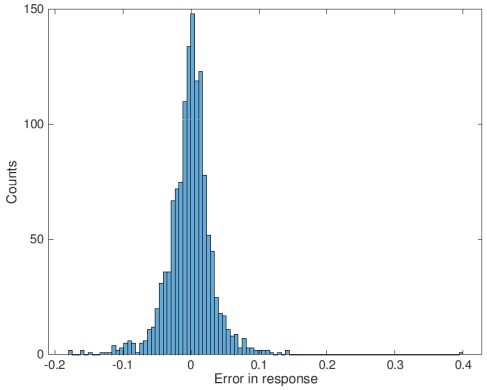


(j) Histogram of the error in the construction of the eleventh mode coefficients. We have $R^2 = 0.368$.

Figure 4.13: Histograms of the error in the construction of the tenth and eleventh mode.

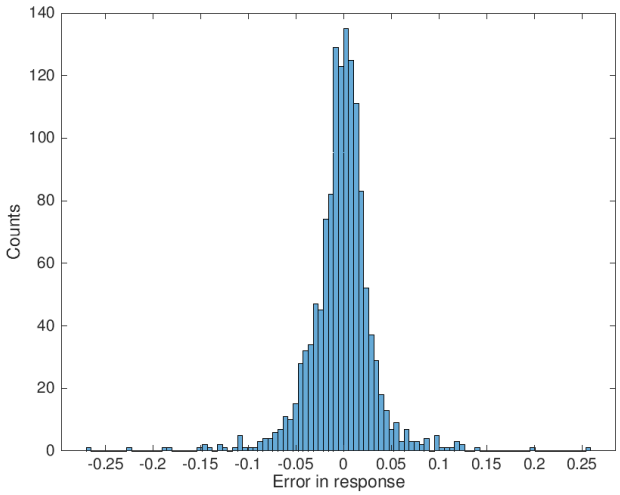


(k) Histogram of the error in the construction of the twelfth mode coefficients. We have $R^2 = 0.831$.



(l) Histogram of the error in the construction of the thirteenth mode coefficients. We have $R^2 = 0.677$.

Figure 4.13: Histograms of the error in the construction of the twelfth and thirteenth mode.



(a) Histogram of the error in the construction of the fourteenth mode coefficients. We have $R^2 = 0.668$.

First of all, we notice that all histograms are centered around zero, and nearly all drop to zero before the error reaches -0.1 or 0.1. Consequently we have that the error is only one order of magnitude smaller than the coefficients values. This is a result that we see in the R^2 values as well, the best R^2 value obtained is $R^2 = 0.948$ for mode two, while the worst that we obtained is $R^2 = 0.368$ for the eleventh mode. When the R^2 is that low, there is no point talking about a fit since practically, there is none. Probably, this low value for R^2 can be explained by a combination of the oscillatory behaviour of the coefficients and the fact that we lose information due to the SVD on the interpolation matrix. We shall now see what the effects on the dose distributions is.

4.3.2. CONSTRUCTION OF THE DOSE DISTRIBUTION MATRIX

We shall again confine ourselves to only some special cases, being the nominal scenario, the two worst reconstructions and the case with maximum error scenario. We shall start with the nominal scenario

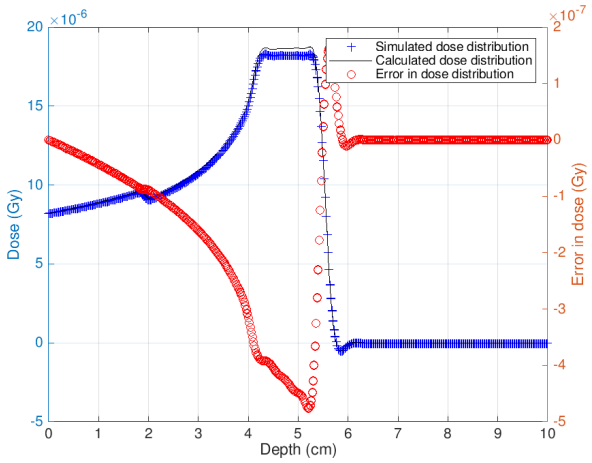


Figure 4.15: Nominal scenario with error $\sigma_t = 0$, $CT_{Tumor} = 0\%$ and $CT_{Tissue} = 0\%$. Dashed line shows the error on the right axis. Error is defined as simulated dose minus constructed dose. $R^2 = 0.999$

We note that for the first time the nominal scenario is not approximated with $R^2 = 1$. Even though it is really close, we can argue that this value is determined for 40% by zeros (6 cm to 10 cm) and thus possibly draws a wrong image. We see that we our dose reconstruction is larger than the simulated dose distribution. In practise this would mean that a patient receives a higher dose than calculated. We also notice, since the error does not integrate to 0, that energy is not conserved in the nominal scenario! However, we see that the drop off by the dose is still present in the same region.

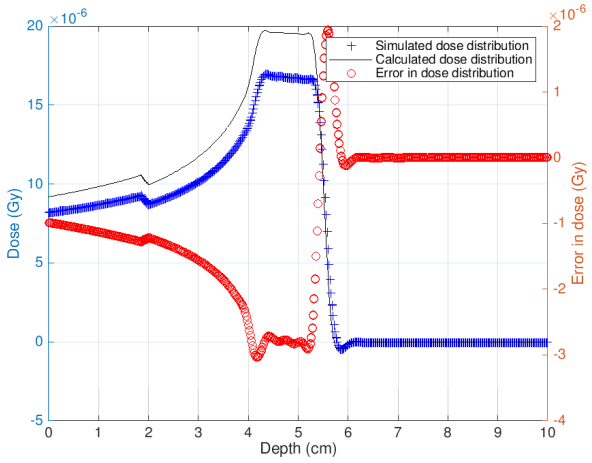


Figure 4.16: Scenario with error $\sigma_t = 0.015$, $CT_{Tumor} = 3\%$ and $CT_{Tissue} = 3\%$. Dashed line shows the error on the right axis. Error is defined as simulated dose minus constructed dose. $R^2 = 0.948$.

From this distribution we see that the reconstruction of the dose distribution matrix did not go too well. However, we still get $R^2 = 0.948$. This very high value for this graph can be explained by two factors. At first, we have the zeros from 6 to 10 cm. Secondly, the shape of the graph is correct. It would probably be better, for all results, to look at the R^2 value until 6cm. For the first time, we do notice a very clear systematic difference between the two dose distributions. The order of the error in the dose distribution is in the same order as the simulated dose distribution! The Hermite polynomial was unsuccessful in reconstructing the V matrix for this specific error scenario. We will now look at a scenario in which the fit is spot on:

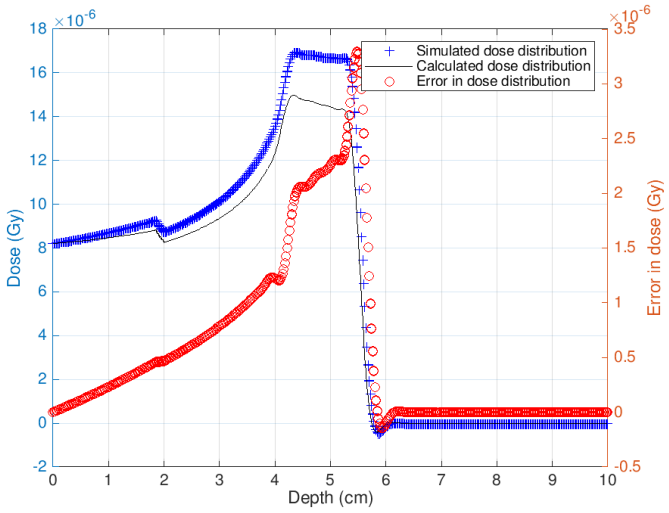


Figure 4.17: Scenario with error $\sigma_t = 0.03$, $CT_{Tumor} = 4\%$ and $CT_{Tissue} = 6\%$. Dashed line shows the error on the right axis. Error is defined as simulated dose minus constructed dose. $R^2 = 0.973$.

The dose distribution in figure 4.17 corresponds to the distribution with the highest error value, not the same as the least R^2 value. We see that our calculated dose distribution falls off too early, which in practise would result in a lower tumor dose, but no extra harm for the tissue behind the tumor. It should be noted that the error scenario $\sigma_t = 0.03$, $CT_{Tumor} = 4\%$ and $CT_{Tissue} = 6\%$ was a data point to fit the Hermite polynomial on. Thus, we would expect the fit to be very good on this scenario. However, the maximum error scenario is also a data point for the fit of the Hermite polynomial and we see that the fit there is not as good as was expected:

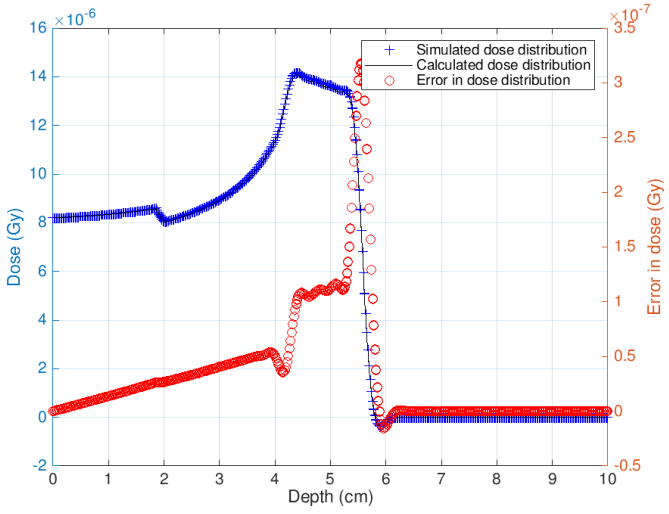


Figure 4.18: Scenario with maximum error $\sigma_t = 0.05$, $CT_{Tumor} = 10\%$ and $CT_{Tissue} = 10\%$. Dashed line shows the error on the right axis. Error is defined as simulated dose minus constructed dose. $R^2 = 1$

We notice that the error is in the order of magnitude of about 10^{-8} except for the dose drop off region where it is in the order 10^{-7}. Overall, the two graphs are rather similar and also result in an $R^2 = 1$. The only discrepancy seems to be in the tumor region, in which the simulated dose is higher than the constructed dose.

Overall, the dose distributions have R^2 close to 1, but we have seen in figure 4.16 that this does not always represent a good image. If we apply the condition as in Algorithm 1, that is all the voxels should be approximated to within 1% precision, we obtain that 70% of the voxels in the dose distribution matrix is accepted.

5

CONCLUSION

We can conclude that fitting a Hermite polynomial to multidimensional simplices in order to find a function that relates an error scenario to a dose distribution, provides a very promising method in proton therapy. For three dimensional errors we have encountered that a Singular Value Decomposition on the Hermite interpolation matrix might have caused the results to look better than they really are. We have seen fits with $R^2 = 1$, or at least very close to one, that graphically do not look particularly good.

We can conclude too that the methods described by Kasbergen about the adjoint theory for proton therapy also work with the boundary conditions given in this project. The adjoint flux is calculated correctly and, with the adjoint flux, the response used in this project can be properly calculated. As a matter of fact, the sensitivities can even be calculated correctly for the responses used in this project. The definition of the right singular vectors as a response in the desired form is very well suited for adjoint and response calculations.

Furthermore, we can also draw the conclusion that the Reduced Order Model method described by Van Galen (Van Galen [1]) works for more dose distributions than he initially tested, although it should be mentioned that he analyzed a full 3D patient. The left singular vectors contain the modes that are needed to reconstruct every dose distribution, and yet we do not need them all in order to achieve an accuracy specified by ourselves.

Since our goal was to improve the results already obtained in the function fitting, we shall compare our results with Van Galen's ([1, p. 27]). We note that the surfaces in 2 dimensions in the results by Van Galen occasionally occur outside the acceptable range of $[-1,1]$ (The results must be in this range in order to preserve orthonormality). We see that the results in this project do occur in this range only for 1 and 2 dimensional errors. As a matter of fact, the lowest R^2 value obtained in this project for errors in 2 dimensions is $R^2 = 0.996$. We conclude that Hermite interpolation provides a better solution to dose reconstruction that regression on the right singular vectors for errors in 1 and 2 dimensions. For the errors in 3 dimensions, we see that Van Galen needs 17 orders of the ROM to achieve a voxel acceptance percentage of 99%, where in this project 70% of the voxels are accepted with only 14 modes of the ROM. We conclude that Hermite interpolation in three dimensions is not as good as it is in two or one dimensions.

A recommendation for further research is that the Hermite interpolation in higher dimensions is calculated with more care. That is, a recipe should be found in order to equal the number of equations to the number of terms in the polynomial. A possible solution might be to fit a polynomial to the entire data set, rather than to simplices as a subset of the data set.

Another recommendation is to test the interpolation method on an irregular grid rather than the regular grid used in this project.

BIBLIOGRAPHY

[1] J. W. Van Galen, *Reduced order modelling methodologies for proton therapy applications,* (2018), Delft University of Technology.

[2] M. Kasbergen, *Adjoint methodologies for proton therapy,* (2018), Delft University of Technology.

[3] H. Paganetti, *Proton Therapy Physics* (Boca Raton: CRC press, 2012) First Edition.

[4] D. Dang, *Dose-depth curve of photons vs. protons,* https://physics.stackexchange.com/questions/169665/dose-depth-curve-of-photons-vs-protons (2015).

[5] D. Davino, *Theory, Design and Tests on a Prototype Module of a Compact Linear Accelerator for Hadrontherapy*, Ph.D. thesis, Università degli Studi del Sannio (2016).

[6] C.-M. Charlie Ma and T. Lomax, *Proton and carbon ion therapy* (Boca Raton: CRC press, 2012) First Edition.

[7] S. B. Uilkema, *Proton Therapy Planning using the SN Method with the Fokker-Planck Approximation*, Master's thesis, Delft University of Technology (2012).

[8] D. Lathouwers, *PHANTOM_SN,* https://bitbucket.org/dlathouwers/proton_sa/src/master/ (2019).

[9] D. Kuzmin, *Galerkin finite element method,* https://www.mathematik.uni-dortmund.de/~kuzmin/cfdintro/lecture7.pdf (2007).

[10] H. S. Abdel-Khalik, Y. Bang, and C. Wang, *Overview of hybrid subspace methods for uncertainty quantification, sensitivity analysis,* Elsevier (2012).

[11] C. Vuik, F. J. Vermolen, M. B. Van Gijzen, and M. J. Vuik, *Numerical Methods for Ordinary Differential Equations* (Delft Academic Press, 2016) pp. 18–20.

[12] The MathWorks Inc., *Delaunay Triangulation - MATLAB & Simulink - MathWorks Benelux,* https://nl.mathworks.com/help/matlab/math/delaunay-triangulation.html (2019).

[13] *Barycentric coordinate system - Wikipedia,* https://en.wikipedia.org/wiki/Barycentric_coordinate_system (2019).

[14] Sourceforge, *Hounsfield Unit - GDCM Wiki,* http://gdcm.sourceforge.net/wiki/index.php/Hounsfield_Unit (2019).

[15] The MathWorks Inc., *Matlab r2018b,* (2018).

[16] W. Schneider, T. Bortfeld, and W. Schlegel, *Correlation between ct numbers and tissue parameters needed for monte carlo simulations of clinical dose distributions,* Physics in Medicine and Biology (1999), Volume 45, Number 2.

A

APPENDIX A

In this appendix we state the code that is used to determine the weights for creating a Spread Out Bragg Peak from e few pristine peaks. The output is a vector **w** containing the weights.

```
breaklines
%% init

table = readtable('results.txt');
num_res = height(table)/1000;

for i = 1:num_res
    doses(:,i) = table{((i-1)*1000+1):(1000*i),2};
end

mesh = table{1:1000,1};

xmin = 4.2;
xmax = 5.3;

opt_dose = 1;

coef = zeros(size(doses(1,:)));

%% Calc

for beam = 1:numel(doses(1,:))

    peak = max(doses(:,beam));
    mesh_value = mesh(doses(:,beam) == peak);
    if ((mesh_value - max_dist < xmax) & (mesh_value + max_dist > xmin))
        SOBP = opt_dose - sum((coef.*doses)');
    else
        SOBP = zeros(size(doses(:,1)));
    end

    try
        coef(beam) = max(SOBP(mesh == mesh_value))/peak;
    end

    coef(isnan(coef)) = 0;
end

fun1 = @(w) opt_dose - sum(((coef ~= 0).*w.*(doses(mesh >= xmin & mesh <= xmax,:)))');
fun2 = @(w)sum(abs(fun1(w)));

options = optimset('MaxFunEvals',10000,'MaxIter',10000);
w = fminsearch(fun2,coef,options);
w = abs(w);

w = (w.*(coef~=0))/max(w);
```

Figure A.1: MATLAB code used for finding the weights to create a SOBP from pristine peaks

B

APPENDIX B

All the code in this appendix is for an error in three dimensions, the cases for two and one dimension begin similar.

REDUCED ORDER MODEL

The following piece of MATLAB code shows how the SVD is done on the dose distribution matrix. The results are the left singular vectors U, the singular values S, the right singular values V and the modes that are needed as an input for Fortran.

```
breaklines
%% init

table = readtable('results3.txt');

num_res = height(table)/1000;
error_dim = [6,6,6];

accep = 0;

mesh = table{1:1000,1};

doses = reshape(table{:,2},1000,[]);

%% calc

[U,S,V] = svd(doses,'econ');

size_d = size(doses);

j=0;
i = size_d(2);
while j<size_d(2)
    j = j+1;
    perc = abs(U(:,1:j)*S(1:j,1:j)*(V(:,1:j)')-doses)/max(doses(:,1));
    tol = sum(sum(perc < 1e-5));
    if ((tol >= numel(doses)) && (j <= i))
        i = j;
    end
    accep(j) = tol;
end

cons_doses = U(:,1:i)*S(1:i,1:i)*(V(:,1:i)');

% create modes for fortran

s=diag(S);

for L = 1:i
    mode(:,:,L) = [U(1:2:end,L)/s(L) U(2:2:end,L)/s(L)];
end
```

Figure B.1: MATLAB code used for finding the weights to create a SOBP from pristine peaks

COMPUTING THE INTERPOLATION POLYNOMIAL

```
breaklines
table=readtable('sensitivities3.txt');
table=table{:,:};

responses = reshape(table(:,1),6,6,6,[]);
sens_sigma = reshape(table(:,2),6,6,6,[]);
sens_CT = reshape(table(:,3),6,6,6,[]);
sens_CT2 = reshape(table(:,4),6,6,6,[]);

terms = [1:3 5:9 11:14 16:19]; % 'terms' contains the terms that are used in the polynomial

for r = 1:i      % 'i' is the number of modes needed for reconstruction
    C(:,:,r) = CompInter3(X,Y,Z,responses(:,:,:,r),sens_sigma(:,:,:,r),...
    sens_CT(:,:,:,r),sens_CT2(:,:,:,r),terms);
    V_temp = plotPoints3(X,Y,Z,X2,Y2,Z2,C(:,:,r),terms);
    V_cons(:,r) = reshape(V_temp,[],1);
end

doses_cons = U(:,1:i)*S(1:i,1:i)*V_cons';    % Matrix containing the constructed dose distribution matrix
```

Figure B.2: MATLAB code used for the computation of the Hermite interpolation polynomial for all modes

NON STANDARD FUNCTIONS USED IN MATLAB CODE B.2

```
breaklines
function [C] = CompInter3(X,Y,Z,F,Fx,Fy,Fz,terms)
    TRI = delaunayn([reshape(X,[],1), reshape(Y,[],1), reshape(Z,[],1)]);
    TRIsize = size(TRI);

    for m = 1:TRIsize(1)
        tri = TRI(m,:);
        [c] = Hermite3(X(tri),Y(tri),Z(tri),F(tri),Fx(tri),Fy(tri),Fz(tri),terms);
        C(m,1:numel(c)) = c;
    end
end
```

Figure B.3: MATLAB code used for the computation of the Hermite interpolation polynomial for all simplices per mode

```
breaklines
function [P] = Hermite3(X,Y,Z,f,fx,fy,fz,terms)
    index2 = 1;
    for n = 1:4
        index2 = 1;
        for px = 0:3
            for py = 0:(3-px)
                for pz = 0:3-px-py
                    H(n,index2)     = X(n)^px * Y(n)^py * Z(n)^pz;
                    H(n+4,index2)   = px*X(n)^(px-1) * Y(n)^py * Z(n)^pz;
                    H(n+8,index2)   = X(n)^(px) * py * Y(n)^(py-1) * Z(n)^pz;
                    H(n+12,index2)  = X(n)^px * Y(n)^py * pz * Z(n)^(pz-1);
                    index2 = index2 + 1;
                end
            end
        end
    end

    ftot = [reshape(f,1,[]),reshape(fx,1,[]), reshape(fy,1,[]), reshape(fz,1,[])]';

    H(isnan(H)) = 0; % This line is necessary since we can have 0^{-1} in the matrix

    [u,d,v] = svd(H(:,terms),'econ');
    s = diag(d);
    s = s(s > max(s)*10^-12);
    u = u(:,1:numel(s));
    di = eye(numel(s))./s;
    v = v(:,1:numel(s));
    P = v*di*(u')*ftot;
end
```

Figure B.4: MATLAB code used for the computation of the Hermite interpolation polynomial for a single simplex per mode

```
breaklines
function [value] = plotPoints3(X,Y,Z,U,V,W,C,terms)
    for i = 1:numel(U)
        value(i) = valueInterPoint3(U(i),V(i),W(i),C(findtri3(X,Y,Z,U(i),V(i),W(i)),:),terms);
    end
end
```

Figure B.5: MATLAB code used for plotting (returning) the interpolated coefficient value.

```
breaklines
function R = findtri3(X,Y,Z,x,y,z)
    TRI = delaunayn([reshape(X,[],1), reshape(Y,[],1), reshape(Z,[],1)]);
    [a,b] = findmin3(X,Y,Z,x,y,z);
    [R,C] = find(TRI == b);
    d=Inf;
    R_temp = R(1);

    % Check in which triangle point is located

    for r = 1:numel(R)
        L = [X(TRI(R(r),:)); Y(TRI(R(r),:)); Z(TRI(R(r),:)); ones(1,4)]\[x;y;z;1];
        if sum((L < 0) | (L > 1)) == 0
            R_temp = R(r);
            break
        end
    end
    R = R_temp;
end
```

Figure B.6: MATLAB code used for finding the simplex in which data point $(U(i), V(i), W(i))$ is located.

```
breaklines
function [location,TRI_value,dmin] = findmin3(X,Y,Z,x,y,z)
    d = sqrt((X-x).^2 + (Y-y).^2 + (Z-z).^2);
    dmin = min(d,[],'all');
    location = (d == dmin);
    temp1 = find(location);
    location = zeros(size(location));
    ran = randi(numel(temp1),1);
    ran = 1;
    location(temp1(ran)) = 1;
    location = (location == 1);
    loc_x = find(X == X(location));
    loc_y = find(Y == Y(location));
    loc_z = find(Z == Z(location));
    TRI_value = intersect(intersect(loc_x,loc_y),loc_z);
end
```

Figure B.7: MATLAB code used for finding the closest snapshot point with respect to the data point $(U(i), V(i), W(i))$.

```
breaklines
function [f] = valueInterPoint3(x,y,z,C,terms)
    f=0;
    index = 1;
    for px = 0:3
        for py = 0:(3-px)
            for pz = 0:3-px-py;
                X(index) = (x.^(px)) * (y.^(py)) * (z.^(pz));
                index = index + 1;
            end
        end
    end
    f = sum(C.*X(terms));
end
```

Figure B.8: MATLAB code used for finding the value of the interpolated point $(U(i), V(i), W(i))$.

CREATING THE SIMULATED MATRICES IN MATLAB AND THE ERROR MATRICES

```
breaklines
table = readtable('results_BIG3.txt');

doses_big = reshape(table{:,2},1000,[]);

V_big = doses_big'*U(:,1:i)*inv(S(1:i,1:i));
V_big = reshape(V_big,11,11,11,14);

V_cons = reshape(V_cons,11,11,11,14);

V_error = reshape(V_big - V_cons,[],14);
```

Figure B.9: MATLAB code used for finding the simulated V matrix.