



VILNIAUS GEDIMINO
TECHNIKOS UNIVERSITETAS

VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS

FUNDAMENTINIŲ MOKSLŲ FAKULTETAS

MATEMATINIO MODELIAVIMO KATEDRA

Edgaras Šimkus

**MATEMATINIS BOLIUSO PAVIRŠIAUS MODELIS
ELEKTRONŲ SRAUTŲ RADIOTERAPIJOJE**

Mathematical bolus design model in electron beam
radiotherapy

Baigiamasis magistro darbas

Technomatematikos studijų programa, valstybinis kodas 621G16001

Matematikos studijų kryptis

Vilnius, 2018

Vilniaus Gedimino technikos universitetas

Fundamentinių mokslų fakultetas

Matematinio modeliavimo katedra

ISBN ISSN

Egz. sk.

Data-....-....

Technomatematikos studijų programos baigiamasis magistro darbas

Pavadinimas **Matematinis boliuso paviršiaus modelis elektronų srautų radioterapijoje**

Autorius **Edgaras Šimkus**

Vadovas prof. habil. dr. **Raimondas Čiegis**

Kalba

☒ X

lietuvių

☐

užsienio

Anotacija

Šiame baigiamajame magistro darbe nagrinėjamas boliuso paviršiaus sudarymo uždavinys elektronų srautų radioterapijoje. Pagrindinis darbo tikslas – nustatyti tokį boliuso paviršiaus modelį, kuris užtikrintų kritinių kūno organų saugumą, maksimalią spinduliuotės dozę taikinyje ir kaip galima mažesnę spinduliuotės poveikį kitiems kūno audiniams.

Boliuso paviršiaus formavimui sprendžiamas daugiakriteris optimizavimo uždavinys atkaitinimo ir genetiniu algoritmais. Elektronų spinduliuotės dozės pasiskirstymui kūne apskaičiuoti taikomas analitinis Hogstromo algoritmas, kuris remiasi kompiuterinės tomografijos nuotrauka vidinių kūno audinių identifikavimui ir fiziškai išmatuotais, taikomos elektronų spinduliuotės, duomenimis.

Darbo apimtis – 38 p. teksto be priedų, 22 iliustr., 6 lent., 33 bibliografiniai šaltiniai.

Atskirai pridedami darbo priedai.

Prasminiai žodžiai: elektronų srautų radioterapija, boliusas, Hogstromo algoritmas, kompiuterinė tomografija, daugiakriteris optimizavimas, atkaitinimo algoritmas, genetinis algoritmas, „MATLAB“.

Vilnius Gediminas Technical University
Faculty of Fundamental Sciences
Department of Mathematical Statistics

ISBN ISSN
Copies No.
Date-....-....

Master Degree Studies **Technomathematics** study programme Master Graduation Thesis

Title **Mathematical bolus design model in electron beam radiotherapy**

Author **Edgaras Šimkus**

Academic supervisor **Prof. Dr. Habil. Raimondas Čiegis**

Language

☒ X

Lithuanian

☐

Foreign (English)

Annotation

The final master thesis examines problem of mathematical bolus design in electron beam radiotherapy. The goal of this work is to develop a mathematical bolus design that ensures safety of critical body organs, maximum dose distribution in target area and as low as reasonably achievable dose in other body areas.

Multi-objective optimization problem was formulated for optimal bolus design and attempted to solve using simulated annealing and genetic optimization methods. Electron beam dose distribution in body is calculated using Hogstrom algorithm which uses computed tomography scan data to determine internal body structure and physically measured data of used electron beam.

Thesis consist of: 38 p. text without appendixes, 22 pictures, 6 tables, 33 bibliographical entries.

Keywords: electron beam radiotherapy, bolus, Hogstrom algorithm, computed tomography, multi-objective optimization, simulated annealing, genetic algorithm, "MATLAB".

Turinys

ĮVADAS	4
1. LITERATŪROS APŽVALGA	5
1.1. Elektronų srautų spinduliuotės savybės	5
1.2. Elektronų sąveika	7
1.2.1. Elektronų energijos praradimas	7
1.2.2. Elektronų sklaida	8
1.3. Monte Karlo metodai	9
1.3.1. <i>Electron Gamma Shower (EGS)</i> programa	9
1.3.2. Macro Monte Karlo algoritmas	10
1.4. Hogstromo algoritmas	14
2. TIRIAMOJI DALIS	19
2.1. Hogstromo algoritmo pritaikymas	19
2.1.1. Duomenys	19
2.1.2. Vėduoklės linijų tinklėlis	20
2.1.3. Spinduliuotės dozės pasiskirstymas vandenyje	22
2.1.4. Spinduliuotės dozės pasiskirstymas paciento kūne	22
2.2. Matematinis boliuso paviršiaus modelis	24
2.2.1. Atkaitinimo algoritmas	26
2.2.2. Genetinis algoritmas	29
IŠVADOS	34
LITERATŪRA	35
PRIEDAI	

Iliustracijų sąrašas

1.	Elektronų spinduliuotės gylio-dozės kreivės vandenyje: vientisa linija pažymėtos kreivės, kurių energija kinta nuo 10 iki 60 MeV, punktyrine kreive pažymėta 5 MeV spinduliuotė (Hogstrom <i>et al.</i> 2016)	6
2.	Daugialapio kolimatoriaus sistema (Romeijn <i>et al.</i> 2005)	6
3.	Boliuso schema (Khanas <i>et al.</i> 2003)	7
4.	δ -spindulių susidarymas (Keall, 1996)	8
5.	Fotono išspinduliavimas (Keall, 1996)	8
6.	Lokalių simuliacijos schema (Zankowski <i>et al.</i> 2004)	11
7.	Kompiuterinės tomografijos nuotraukos apdorojimas (Zankowski <i>et al.</i> 2004)	12
8.	„Eclipse“ algoritmo schema (Zankowski <i>et al.</i> 2004)	13
9.	Elektronų spindulio sklaidos schema (Hogstromas <i>et al.</i> 1981)	15
10.	Elektronų gylio-dozės kreivė pavaizduota kartu su praktiniu diapozonu (Strydom <i>et al.</i> 2012)	16
11.	17 MeV elektronų spindulio izodozinės linijos vandenyje (Hogstrom <i>et al.</i> 1980)	18
12.	6 MeV energijos elektronų spinduliuotės gylio-dozės pasiskirstymas vandenyje	20
13.	Paciento galvos kompiuterinės tomografijos nuotrauka	21
14.	Vėduoklės linijų tinklelis	21
15.	Vandens fantomas	22
16.	Elektronų spinduliuotės dozė vandens fantome	23
17.	Nagrinėjama kompiuterinės tomografijos nuotraukos sritis	23
18.	6 MeV elektronų srauto dozės pasiskirstymas paciento kūne	24
19.	Elektronų srauto dozės pasiskirstymas, kai boliuso formavimui naudotas atkaitinimo metodas ir (2.2) tikslo funkcija	28
20.	Elektronų srauto dozės pasiskirstymas, kai boliuso formavimui naudotas atkaitinimo metodas ir (2.3) tikslo funkcija	29
21.	Elektronų srauto dozės pasiskirstymas, kai boliuso formavimui naudotas genetinis algoritmas ir (2.2) tikslo funkcija	32
22.	Elektronų srauto dozės pasiskirstymas, kai boliuso formavimui naudotas genetinis algoritmas ir (2.3) tikslo funkcija	33

Lentelių sąrašas

1.	<i>Eclipse</i> Monte Karlo algoritmo skaičiavimo laikas su skirtingomis energijos ir skirtingais nagrinėjamo lauko dydžiais (Chamberland <i>et al.</i> 2015)	13
2.	Stabdomosios ir sklaidomosios gebų reikšmės skirtinguose audiniuose	20
3.	Boliuso poveikis dozės pasiskirstymui naudojant atkaitinimo metodą (1)	28
4.	Boliuso poveikis dozės pasiskirstymui naudojant atkaitinimo metodą (2)	29
5.	Boliuso poveikis dozės pasiskirstymui naudojant genetinį algoritmą (1)	31
6.	Boliuso poveikis dozės pasiskirstymui naudojant genetinį algoritmą (2)	32

ĮVADAS

Hogstromo algoritmas naudojamas apskaičiuoti elektronų srauto dozės pasiskirstymą nehomogeniškoje terpėje. Remiantis algoritmo rezultatais išskiriamos spinduliuotės paveiktos sritys ir įvertinamas kūnui suteiktas energijos kiekis. Pagrindinis nagrinėjamo uždavinio tikslas – nustatyti tokį boliuso paviršiaus modelį, kuris užtikrintų kritinių kūno organų saugumą, maksimalią spinduliuotės dozę taikinyje ir kaip galima mažesnę spinduliuotės poveikį kitiems kūno audiniams.

Aprašytas uždavinys ypatingai aktualus medicinos fizikoje, radiologijos srityje, kur nuo vėžinių susirgimų kenčiantys pacientai gydomi išorine elektronų spinduliuote. Nepaisant gydymo metodų ir technologijų įvairovės radiologijos srityje, gydymas išorine spinduliuote lieka viena iš dažniausiai atliekamų procedūrų gydymo plano vykdymo metu.

Šiuolaikiniuose elektronų srauto energijos skaičiavimo technologijose įprastai naudojami Monte Karlo metodai, kurie yra realizuoti tokiuose komerciniuose programiniuose paketuose kaip „Varian“, „Accuray“ ir kt. Verta paminėti, kad šioje srityje egzistuoja atviro kodo programinė įranga tokia, kaip, pavyzdžiui, „EGS4“. Monte Karlo metodai laikomi tiksliais – gautų rezultatų nuokrypis nuo išmatuotų skiriasi nereikšmingai mažai, tačiau šie metodai reikalauja ypatingai daug skaičiavimų, jie nagrinėja kiekvieną elementariąją dalelę, jos trajektoriją ir energiją. Tokie skaičiavimai reikalauja daug laiko, todėl ieškoma būdų kaip bent keletą kartų padidinti skaičiavimo laiką, tačiau neprarasti Monte Karlo metodo tikslumo.

Siekiant pagreitinti rezultatų apskaičiavimo laiką, taikomas analitinis Hogstromo algoritmas, kur vietoje atskirų elektronų nagrinėjami pieštukiniai spinduliai (angl. *pencil beam*), sudaryti iš elektronų dalelių pluošto. Papildomai, siekiant sutelkti spindulio koncentraciją ir apsaugoti audinius nuo perteklinio spinduliuotės poveikio taikomi audinių kompensatoriai – boliusai.

Pagrindiniai uždaviniai:

- Pateikti Hogstromo ir Monte Karlo algoritmų teorinę apžvalgą;
- Realizuoti Hogstromo algoritmą naudojant „MATLAB“ programinę įrangą;
- Pritaikyti Hogstromo algoritmą naudojant kompiuterinės tomografijos nuotrauką;
- Sudaryti matematinį boliuso paviršiaus modelį naudojant atkaitinimo (angl. *simulated annealing*) ir genetinį optimizavimo algoritmus.

1. LITERATŪROS APŽVALGA

1.1. Elektronų srautų spinduliuotės savybės

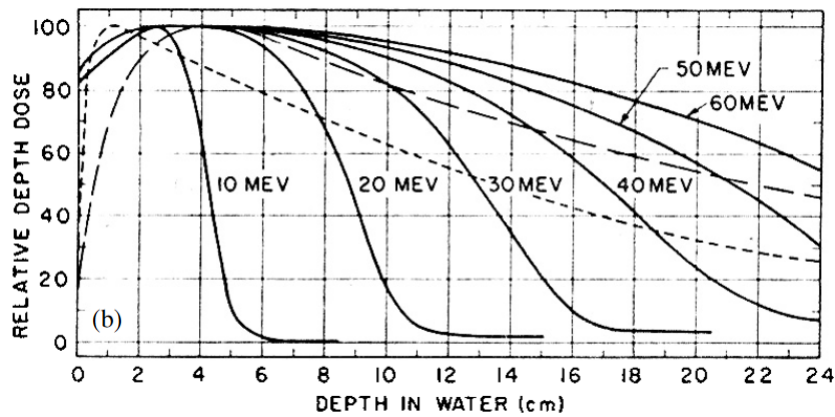
Elektronų srauto pluoštas pasižymi svarbiomis savybėmis, reikalingomis radioterapijoje: (i) dozės pasiskirstymas audiniuose yra tolydus; (ii) elektronų energijos kiekis, prarandamas dalelei keliaujant skirtingais audiniais, kinta nežymiai; (iii) spinduliuotės prasiskverbimo gylis gali būti kontroliuojamas keičiant pradinį energijos dydį ir taikant audinio kompensatorius – boliusus. (Kim *et al.* 2015).

Be šių svarbių savybių, elektronų spinduliuotė taip pat pasižymi sparčiu energijos praradimu pasiekus santykinį energijos kiekį tarp 90 % ir 100 % lyginant su maksimaliu spindulio energijos kiekiu centrinės spindulio ašies atžvilgiu. Ši savybė ypatingai svarbi siekiant spinduliuote paveikti arti kūno paviršiaus esančius audinius, taip pat – kuo mažesniu energijos kiekiu paveikti gilesnius kūno audinius.

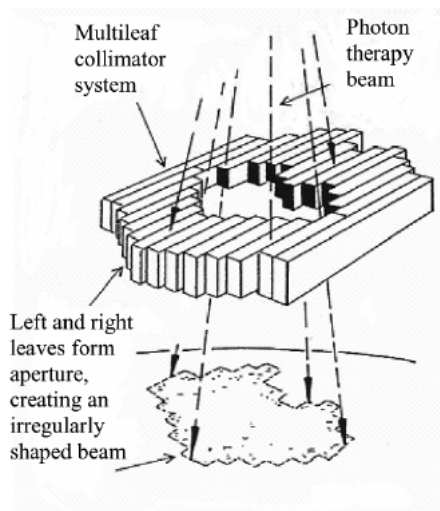
Elektronų srauto spinduliuotė naudojama siekiant paveikti audinius, esančius iki 6 cm gylyje, tačiau kartu įvedamas papildomas apribojimas – pradinis spinduliuotės energijos dydis turi būti nuo 6 iki 20 megaelektronvoltų (MeV). 1 paveiksle pateiktas grafikas, kuriame pavaizduotos skirtingų spinduliuotės energijų santykinės gylis-dozės kreivės. Horizontalioje ašyje žymimas gylis centimetrais, vertikalioje – santykis tarp spinduliuotės suteikiamos dozės ir maksimalios dozės centrinės ašies atžvilgiu. Paveiksle matoma, kad kai spinduliuotės energija yra 10 arba 20 MeV, atitinkama kreivė pasižymi staigiu kritimu, tačiau kitiems energijos kiekiams ši savybė nėra išsaugoma – 5 MeV ir 30-50 MeV energijos spinduliuotės kreivės taip sparčiai energijos nepraranda; tai reiškia, kad spinduliuotė gali stipriai paveikti audinius, esančius giliau nei 6 cm, tai dažnu atveju nėra pageidautina, kadangi tam naudojama, kitokio tipo spinduliuotė, pavyzdžiui, fotonų, kurios savybės yra tinkamesnės gilesniems audiniams.

Sritys, kurias norima paveikti jonizuojančia elektronų spinduliuote, įprastai yra nestandartinių formų, todėl priemonės, kuriomis formuojama spinduliuotės pluošto forma, ypatingai svarbios. Fotonų spinduliuotės atveju yra sukurti specialūs įrenginiai vadinami daugialapiais kolimatoriais (angl. *multileaf collimator*), šis prietaisas įtaisytas kartu su jonizuojančią fotonų spinduliuotę generuojančiu įrenginiu – tiesiniu dalelių greitintuvu. Daugialapio kolimatoriaus schema pateikiama 2 paveiksle.

Elektronų spinduliuotės atveju šis įrenginys tinkamas tik su tam tikromis išlygomis, kadangi elementariosios dalelės yra išsklaidomos ir spinduliuotės pasiskirstymas gali tapti neto-



1 pav. Elektronų spinduliuotės gylio-dozės kreivės vandenyje: vientisa linija pažymėtos kreivės, kurių energija kinta nuo 10 iki 60 MeV, punktyrine kreive pažymėta 5 MeV spinduliuotė (Hogstrom *et al.* 2016)

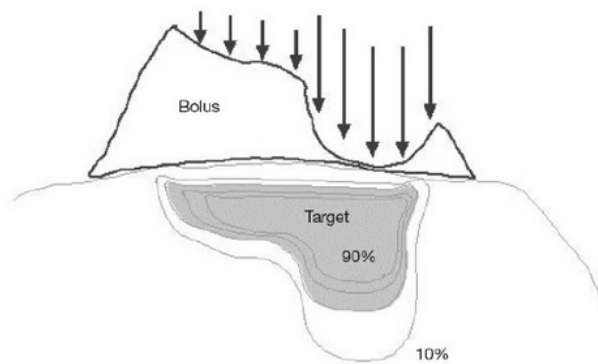


2 pav. Daugialapio kolimatoriaus sistema (Romeijn *et al.* 2005)

lygus. Nepaisant to, ieškoma būdų efektyviai moduluoti elektronų spinduliuotę, pavyzdžiui, A. Eldibas savo straipsnyje (Eldibas *et al.* 2009) siūlo du būdus: modifikuoti praktikoje naudojamo tiesinio dalelių greitintuvo galvelę, orą jos viduje pakeičiant heliu arba naudoti papildomus spindulius, kuriais būtų moduluojama elektronų spinduliuotė.

Įprastai naudojamas metodas elektronų spinduliuotei moduluoti – taikyti audinio kompensatorius – boliusus (angl. *bolus*). Tai yra iš polimerų pagamintas produktas, kurio savybės elektronų spinduliuotės atžvilgiu yra panašios į vandens savybes. Boliuso schema pateikiama 3 paveiksle.

Pagrindiniai tikslai, kurių siekiama naudojant boliusus yra šie: (i) apsaugoti jautrius audinius, kurių negalima pažeisti jonizuojančia spinduliuote ir (ii) išlyginti spinduliuotės lauko formą nagrinėjamame kūne.



3 pav. Boliuso schema (Khanas *et al.* 2003)

Naudojant, šiuo metu vis labiau prieinamą technologiją – 3D spausdintuvus, kiekvienam kūnui, per priimtina laiką, gali būti sukurtas individualus boliusas.

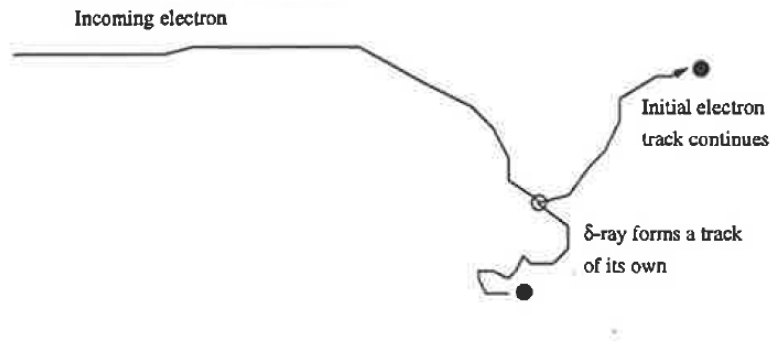
1.2. Elektronų sąveika

1.2.1. Elektronų energijos praradimas

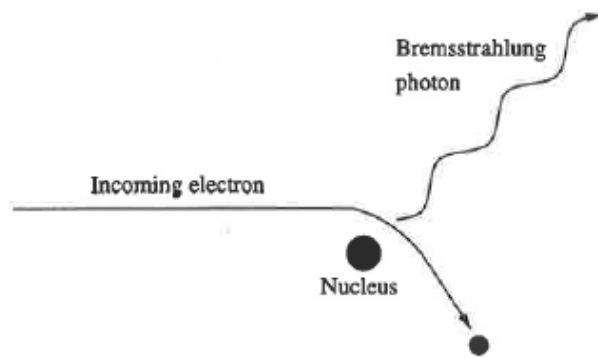
Radiologijos srityje naudojamas elektronų srautas, energiją praranda iš esmės dviem būdais: dėl elementariųjų dalelių susidūrimo nuostolių (angl. *collisional stopping power*) ir dėl dalelių išspinduliuojamos energijos (angl. *radiative stopping power*). Šie procesai yra neterministiniai, todėl skirtingoms dalelėms turi skirtingą įtaką, tai reiškia, kad net ir tos pačios energijos ir identiškoje pozicijoje esančių elementariųjų dalelių trajektorijos bus skirtingos.

Susidūrimo nuostoliais vadinamas energijos praradimas, kai elementarioji spinduliuotės srauto dalelė juda nagrinėjamu kūnu ir dėl sąveikos su kūne esančiais atomais ir aplink skriejančiais elektronais praranda energiją dėl jonizacijos ir dėl kitų dalelių sužadavimo. Taip pat svarbu paminėti Miolerio efektą – didelės energijos elektronai gali susidurti su pririštais elektronais taip stipriai, kad pastarieji palieka atomą prie kurio buvo prisirišę ir kūnu juda toliau kartu su laisvuju elektronu. Šio proceso schema pateikiama 4 paveiksle. Elektronai palikę atomą vadinami δ -spinduliais. (Keall, 1996).

Energijos išspinduliavimas yra būdingas dalelėms, kurių energija yra itin didelė. Kai pastaroji dalelė juda kūnu ir panteka į atomo aplinką, dėl Kūlono traukos, jos trajektorija pasikeis nukrypdoma atomo link, šio proceso metu išspinduliuojamas fotonas (angl. *bremsstrahlung*), proceso schema pateikiama 5 paveiksle. Šis efektas ypatingai daug įtakos turi, kai nagrinėjama medžiaga turi aukštą atominį skaičių. (Keall, 1996).



4 pav. δ -spindulių susidarymas (Keall, 1996)



5 pav. Fotono išspinduliavimas (Keall, 1996)

1.2.2. Elektronų sklaida

Elektronų sklaida pasireiškia šoniniu elektronų spindulio pasiskirstymu nagrinėjamame kūne. Dėl šio efekto spinduliuotės suteikiama dozė pasiskirsto didesniame kūno plote, kartu sumažėja spindulio skvarba į gilesnius kūno audinius. Elektronų sklaida ir energijos praradimas yra tarpusavyje susiję procesai, kadangi fotonų spinduliavimas ir δ -spinduliai sukelia didesnę dalelių sklaidą, tačiau taikomuosiuose algoritmuose šie procesai dažniausiai nagrinėjami kaip nepriklausomi. Atsižvelgiant į skirtingų algoritmų tikslą, išskiriami trys pagrindiniai sklaidos tipai: vienkartinė sklaida (angl. *single scattering*); daugkartinė sklaida (angl. *plural scattering*); kartotinė sklaida (*multiple scattering*). Pieštukinių spindulių algoritmuose dažniausiai taikoma kartotinės sklaidos tipas. (Keall, 1996).

Kartotinė sklaida apima jonizacijos ir sužadinimo procesus ir nagrinėja juos kaip vieną bendrą procesą. Praktikoje dažniausiai taikomos trys skirtingos elektronų kartotinės sklaidos teorijos: Goudsmito ir Saundersono (Goudsmit *et al.* 1940), Moliero ir Fermi-Eyges. Iš pateiktų teorijų, Fermi-Eyges yra dažniausiai taikoma pieštukinių spindulių metoduose.

Fermi-Eyges lygtis aprašanti kartotinę Kūlono sklaidą užrašoma kaip

$$\frac{\partial P(\theta_x, x, z)}{\partial z} = -\theta_x \frac{\partial P}{\partial x} + \frac{T_s(z)}{4} \frac{\partial^2 P}{\partial \theta_x^2}, \quad (1.1)$$

čia $P(\theta_x, x, z)$ – sklaidomųjų dalelių judėjimo z ašies kryptimi tikimybės pasiskirstymo funkcija, z – gylis nagrinėjamame kūne, x – dalelės pozicija horizontalios ašies atžvilgiu, θ_x – dalelės judėjimo kampas $x - z$ ašių atžvilgiu, $T_s(z)$ – tiesinė sklaidomoji geba. Šios lygties sprendinys yra

$$P(\theta_x, x, z) = \frac{1}{\pi \sqrt{A_0 A_2 - A_1^2}} e^{-(A_0 x^2 - 2A_1 x \theta_x + A_2 \theta_x^2)/(A_0 A_2 - A_1^2)}, \quad (1.2)$$

čia $A_i (i = 0, 1, 2)$ – pieštukinio spindulio sklaidos momentai, kurie aprašomi kaip

$$A_i(z) = \frac{1}{2} \int_0^z T_s(z') (z - z')^i dz'. \quad (1.3)$$

Norint įvertinti tikimybę, kad elektronas gylyje z pajudės į šoną per x , nepriklausomai nuo θ_x , sprendinys (1.2) yra integruojamas visiems θ_x ir gaunamas šoninio dalelių srauto tikimybės sprendinys aprašomas kaip

$$P_L(x, z) = \frac{1}{\sqrt{2\pi}\theta_x} e^{\frac{-x^2}{2\theta_x^2}}, \quad (1.4)$$

gautas Gauso tikimybinis skirstinys su vidurkiu 0, amplitude $1/\sqrt{2\pi}\theta_x$ ir standartiniu nuokrypiu θ_x , kur $\theta_x = \sqrt{\frac{1}{2} \int_0^z T_s(z') (z - z') dz'}$. (Keall, 1996).

1.3. Monte Karlo metodai

Monte Karlo metodai sudaro plačią algoritmų klasę, kuri remiasi pakartotiniu atsitiktinių imčių generavimu, šie metodai ypatingai gerai tinka elementariųjų dalelių pernešimui modeliuoti, kadangi jų judėjimas gali būti nuspėtas tik su tam tikra tikimybe. Monte Karlo metodai modeliuoja visus fizikinius, dalelių pernešimo dėsnius, todėl skirtumas tarp šiais metodais gautų rezultatų ir išmatuotų gali būti nereikšmingai mažas, jei skaičiavimams yra skiriama pakankamai daug laiko ir nagrinėjami objektai aprašomi pakankamai tiksliai. Šiuo atveju nagrinėjami objektai yra: elementariųjų dalelių pozicija, kryptis ir energija; nagrinėjamo kūno vidinė struktūra. Toliau aprašomos kai kurios, populiarios Monte Karlo algoritmų programos, dalelių pernešimo uždaviniams spręsti. (Nelson *et al.* 1985).

1.3.1. *Electron Gamma Shower (EGS)* programa

EGS Monte Karlo programa yra skirta elektronų ir fotonų dalelių pernešimui modeliuoti, jis pristatytas 1978 m. ir nuo to laiko išlieka dažniausiai taikomu algoritmu medicinos fizikos

srityje (Tessier *et al.* 2015). *EGS* yra atviros prieigos sistema, kurią galima rasti gamintojo internetinėje svetainėje, šią sistemą sudaro: sisteminis kodas ir naudotojo aprašomas kodas. Naudotojas privalo aprašyti 3 paprogrames:

1. *MAIN* – įvedami simuliacijos parametrai ir gaunami apskaičiuoti rezultatai;
2. *HOWFAR* – aprašoma sistemos ir nagrinėjamo kūno geometrija;
3. *AUSGAB* – nurodama kur ir kaip saugomi gauti rezultatai.

EGS sisteminį kodą sudaro 4 paprogramės:

1. *HATCH* – paruošiami naudotojo pateikti duomenys;
2. *SHOWER* – paruošiama elementariosios dalelės simuliacija su pradine pozicija, kryptimi ir energijos kiekiu;
3. *ELECTR* – modeliuojamas elektronų ir δ -spindulių judėjimas, kuris buvo paruoštas *SHOWER* paprogramės;
4. *PHOTON* – modeliuojamas fotono judėjimas, kuris buvo paruoštas *SHOWER* paprogramės.

Elektrono judėjimo modeliavimas apima tokius fizikinius procesus kaip: *Bhabba* ir *Moller* sklaidos; Moliero sklaida; fotono išspinduliavimas; jonizacijos energijos nuostoliai; pilnas dalelės energijos praradimas. Atskiros dalelės algoritmo metu yra nagrinėjamos tol kol jų energija yra reikšminga ir dalelių pozicija yra nagrinėjamame kūne. Dalelių judėjimas, energija ir sąveika modeliuojama remiantis atsitiktiniais dydžiais gautais iš tikimybinių skirstinių, kurie aprašo jau minėtus fizikinius dėsnius.

1.3.2. Macro Monte Karlo algoritmas

Macro Monte Karlo algoritmas buvo pasiūlytas Maki (Mackie *et al.* 1984) su tikslu – pagreitinti pilną Monte Karlo metodą neprarandant tikslumo. Šiam tikslui pasiekti atsisakoma nagrinėti labai mažus elektrono būsenos pokyčius, vietoje to, nagrinėjami dalelės būsenos pokyčiai didesniuose žingsniuose. Pokyčiai juose apskaičiuojami remiantis dalelių judėjimo tikimybiniais skirstiniais iš *EGS* Monte Karlo programos. Šis algoritmas yra taikomas šiuolaikiniuose komerciniuose programiniuose paketuose, kaip, pavyzdžiui, *Eclipse*, kuris šiuo metu yra naudojamas Lietuvos nacionaliniame vėžio institute.

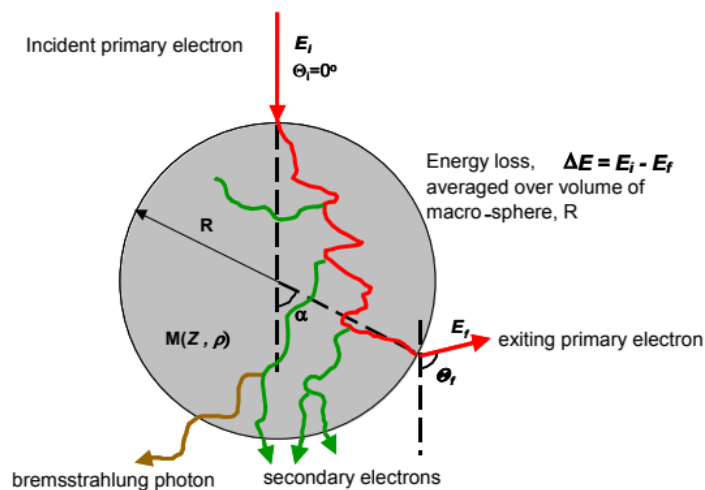
Eclipse programinio paketo elektronų pernešimo algoritmas naudoja Macro Monte Karlo algoritmą, tačiau papildomai, makroskopinis žingsnis čia priklauso nuo kūno vidinės struktūros – žingsnis yra mažas kur vyrauja heterogeniška aplinka ir didelis, kai aplinka homogeniška. Šio algoritmo bendrą skaičiavimo laiką T galima aprašyti

$$T = \tau_{step} \cdot \eta_{step} \cdot N_{hist}, \quad (1.5)$$

čia τ_{step} – vieno žingsnio skaičiavimo laikas, η_{step} – žingsnių kiekis reikalingas pilnam dalelės keliui, N_{hist} – visas naudotų dalelių kiekis. *Eclipse* algoritmas siekia kaip įmanoma greičiau apskaičiuoti τ_{step} ir η_{step} . Pagrindiniai algoritmo etapai išskiriami į: (i) erdvės modelio sudarymas; (ii) lokali simuliacija; (iii) geometrijos apdorojimas; (iiii) globali simuliacija.

(i) Erdvės modelio sudarymas. Šiuo modeliu aprašoma sistemos geometrija – elektronų spinduliuotės šaltinio pozicija, kūno pozicija, nurodomas atstumas tarp kūno ir šaltinio, pateikiami duomenys apie naudojamų kolimatorių savybes dalelių srautui moduluoti ir kiti duomenys.

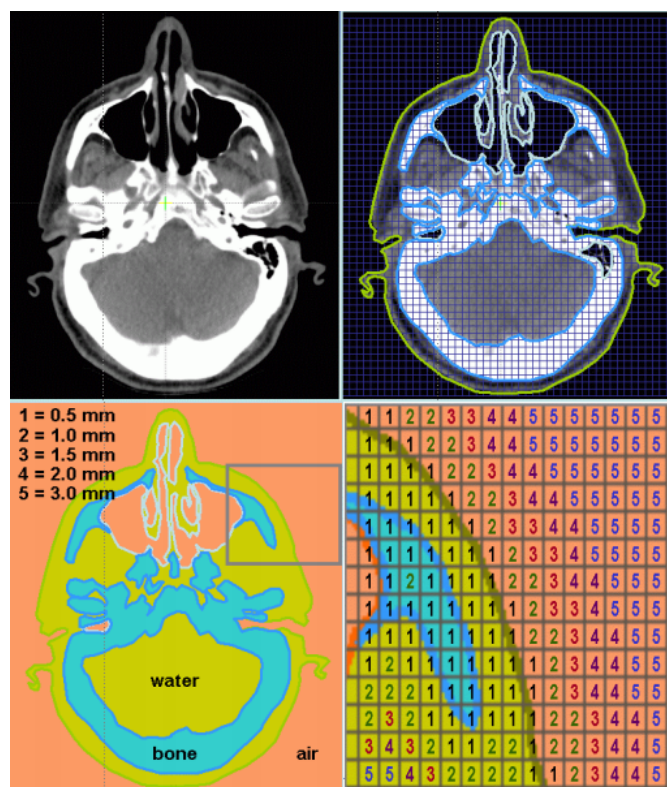
(ii) Lokali simuliacija. Šio etapo metu atliekama bandymų serija, modeliuojant vieno elektrono trajektoriją vienoje makroskopinio dydžio sferoje, šio proceso schema pateikiama 6 paveiksle.



6 pav. Lokali simuliacijos schema (Zankowski *et al.* 2004)

Simuliacijos metu taikomi tikimybiniai skirstiniai aprašantys fizikinius procesus aprašytus 1.2 poskyryje, taip pat nustatomas energijos dydis ir kryptis dalelei paliekant makroskopinę sferą. Sferoje nusėdęs energijos kiekis $E = E_i - E_f - E_\delta$, čia E_i – elektrono energija trajektorijos pradžioje, E_f – elektrono energija trajektorijos pabaigoje, E_δ – energijos kiekis prarastas dėl antrinių dalelių.

(iii) **Geometrijos apdorojimas.** Siekiant užtikrinti algoritmo greitį *Eclipse* iš anksto apdoroja duomenis apie nagrinėjamo kūno vidinę struktūrą – kompiuterinės tomografijos nuotrauką. Apdorojimo metu kiekvienam nuotraukos pikseliui priskiriamas skaičius indukuojantis maksimalų makroskopinės sferos spindulio dydį naudojamą lokaliaje simuliacijoje, šio proceso iliustracija pateikiama 7 paveiksle.

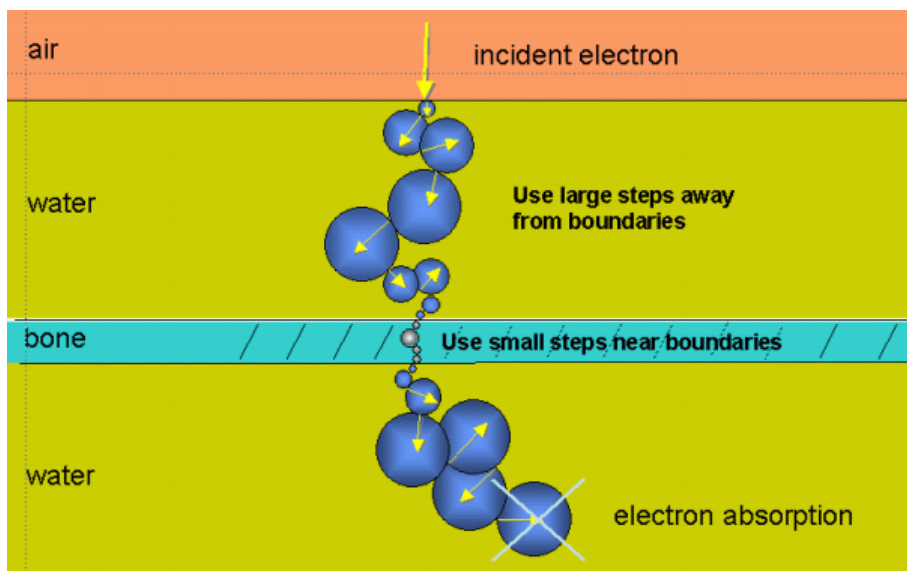


7 pav. Kompiuterinės tomografijos nuotraukos apdorojimas (Zankowski *et al.* 2004)

Aukšta reikšmė indikuoja didelį leidžiamą sferos spindulį, maža reikšmė – mažą spindulį. Indikatoriaus reikšmė priklauso nuo kompiuterinės tomografijos nuotraukos pikselio aplinkos: jei pikselio aplinkoje yra heterogeniškų audinių, tuomet indikatoriaus reikšmė nustatoma maža, o jei homogeniška – didesnė. rrr

(iiii) **Globali simuliacija.** Šis etapas apjungia jau aprašytus (i), (ii) ir (iii) algoritmo procesus: elektrono dalelė juda nagrinėjamu, heterogenišku kūnu iš anksto numatyto dydžio sferomis tol, kol elektrono energija sumažėja iki tokios ribos nuo kurios laikoma, kad jis nebeteri įtakos dozės pasiskirstymui arba palieka nagrinėjamą kūną. Visų etapų apibendrinta schema pateikiama 8 paveiksle.

Dėl Monte Karlo metodų tikslumo, jie dažnai naudojami kitų, analitinių metodų testavimui, tačiau dėl didelių skaičiavimo apimčių ir laiko skaičiavimams atlikti, ieškoma efektyvesnių metodų dalelių pernešimo uždaviniams spręsti. 1 lentelėje pateikiamas *Eclipse* al-



8 pav. „Eclipse“ algoritmo schema (Zankowski *et al.* 2004)

goritmo dozės pasiskirstymo skaičiavimo laikas skirtingos energijos spinduliams ir skirtingo dydžio laukuose.

1 lentelė. *Eclipse* Monte Karlo algoritmo skaičiavimo laikas su skirtingomis energijos ir skirtingais nagrinėjamo lauko dydžiais (Chamberland *et al.* 2015)

Energijos kiekis	Lauko dydis (cm ²)	Skaičiavimo laikas (min)
6 MeV	6 × 6	1,78
	20 × 20	16,95
12 MeV	6 × 6	4,23
	20 × 20	28,25
18 MeV	6 × 6	7,00
	20 × 20	48,00

Elektronų srauto dozės pasiskirstymo apskaičiavimo greitis ypatingai svarbus sudarant optimalų boliuso paviršiaus modelį, kadangi taikomi optimizavimo algoritmai privalo ne vieną kartą perskaičiuoti spinduliuotės dozės pasiskirstymą kūne su skirtingais boliuso modeliais. Yra žinoma, kad analitiniai sprendimo metodai gali pasiekti geresnius rezultatus laiko atžvilgiu, todėl šiame darbe pasirinkta nagrinėti analitinį K. Hogstromo algoritmą (Hogstrom *et al.* 1981), kuris remiasi pieštukinio spindulio algoritmu. Taip pat, galima paminėti kitus analitinius algoritmus, tokius kaip: efektinio gylio metodas (angl. *effective depth*) (Almond *et al.* 1967); pieštukinio spindulio metodas (angl. *pencil beam*); 3-D pieštukinio spindulio

metodas (Kooy *et al.* 1989); pieštukinio spindulio patikslinimo metodas (angl. *pencil beam redefinition*) (Hogstrom *et al.* 1991).

1.4. Hogstromo algoritmas

Hogstromo algoritmu elektronų spindulio pluoštas skaidomas į smulkesnius spindulius vadinamus pieštukiniais spinduliais (angl. *pencil beam*), kurie atstoja pradinį elektronų spindulio pluoštą, pluošto suteikiama dozė apskaičiuojama kaip atskirų pieštukinių spindulių dozės suma. Hogstromo algoritmas papildoma pirmtakų D. Perio ir J. Holto (Perry, Holt, 1980) modelį atsižvelgiant į šiuos aspektus: (i) nagrinėjant kompiuterinės tomografijos (KT) duomenis įvertinami tolydūs duomenų pokyčiai (ii) tolydus elektronų energijos praradimas audiniuose (iii) vienu metu apskaičiuojama heterogeniškų audinių sukeliamą sklaidos ir spindulio pusėsėšlio (angl. *penumbra*) įtaka.

K. Hogstromas aprašo kartotinę Kūlono sklaidą (angl. *multiple Coulomb scattering*) remdamasis Fermi-Eyges aprašyta teorija (Eyges, 1948), taip pat įvertinama Miolerio sklaida, tačiau neatsižvelgiama į elektronų sklaidos pasiskirstymą storuose sluoksniuose, kai išsiskiria antrinis elektronas. Energijos išspinduliavimas (angl. *bremsstrahlung*), kuomet elektronas prarasdamas greitį išskiria fotoną nėra reikšmingas, todėl modelyje jį kompensuoja fiziškai išmatuoti spinduliuotės gylio-dozės duomenys. Tokie kūno geometrijos aspektai kaip kūno kontūras ir oro tarpai kūne įtraukti į algoritmą neišreikštiniu būdu.

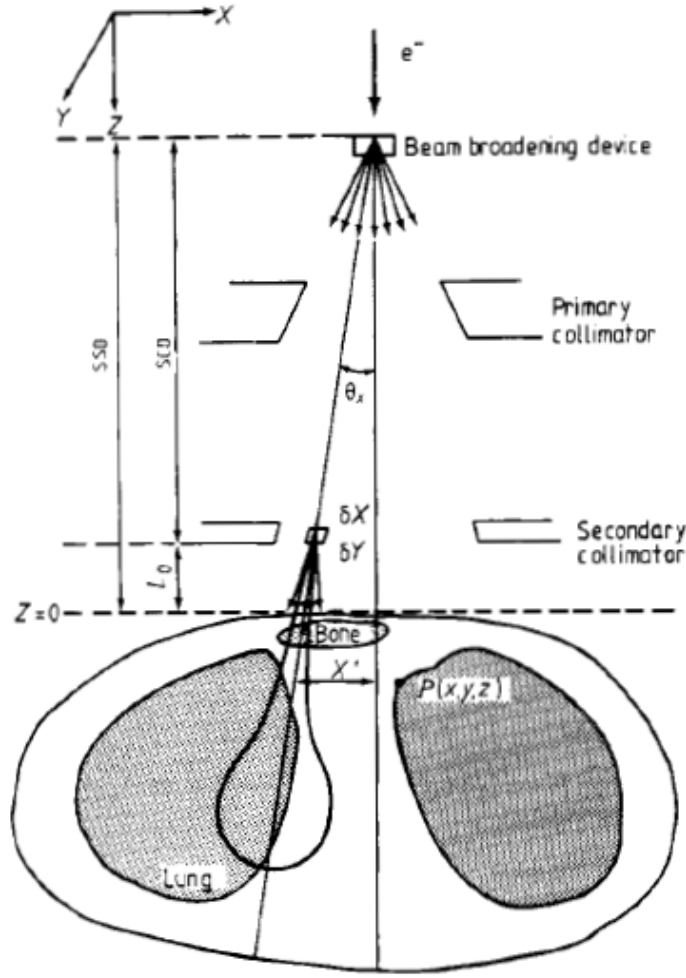
9 paveiksle pateikiama schema vaizduojanti, kaip siauras elektronų spindulys, kurį generuoja tiesinis dalelių greitintuvas, išplečiamas ir suformuojamas naudojant atitinkamus įrenginius iki paskutinio kolimatoriaus.

Pieštukinį spindulį sudaro dalelės, kurios skrieja per nykstamai mažą plotą $\delta X \delta Y$ paskutinio kolimatoriaus aukštyje. Laikoma, kad visų pieštukinį spindulį sudarančių elektronų energija yra vienoda, o jų kampinis nuokrypis žymimas θ_x, θ_y . Norint gauto dozės pasiskirstymą kūne $D(X, Y, Z)$ sumuojami visų pieštukinių spindulių dozės pasiskirstymai naudojant (1.6) formulę:

$$D(X, Y, Z) = \iint_Z S(X', Y') d(X' - X, Y' - Y, Z) dX' dY', \quad (1.6)$$

čia $S(X', Y')$ – pieštukinio spindulio santykinė galia taške (X', Y') , $d(X' - X, Y' - Y, Z)$ – spinduliuotės dozės įnašas taške (X, Y, Z) iš (X', Y') . Pastarasis dydis išskaidomas į dvi komponentes: pieštukinio spindulio centrinės ašies dedamąją komponentę $g(Z)$ ir ne centrinės ašies dedamąją $f(X, Y, Z)$:

$$d(X, Y, Z) = g(Z)f(X, Y, Z). \quad (1.7)$$



9 pav. Elektronų spindulio sklaidos schema (Hogstromas *et al.* 1981)

Laikoma, kad ne centrinės ašies komponentė $f(X, Y, Z)$ lygi horizontaliam dalelių srauto pasiskirstymui ir remiantis Eyges teorija elektronams aprašoma

$$f(X, Y, Z) = \frac{1}{2\pi\sigma_{MCS}^2} e^{-\frac{X^2+Y^2}{2\sigma_{MCS}^2}}, \quad (1.8)$$

kur σ_{MCS} aprašo horizontalų pasiskirstymą

$$\sigma_{MCS}^2 = \frac{1}{2} \int_{-L_0}^Z (Z - Z')^2 \frac{d\sigma_{MCS}^2}{dZ'} dZ', \quad (1.9)$$

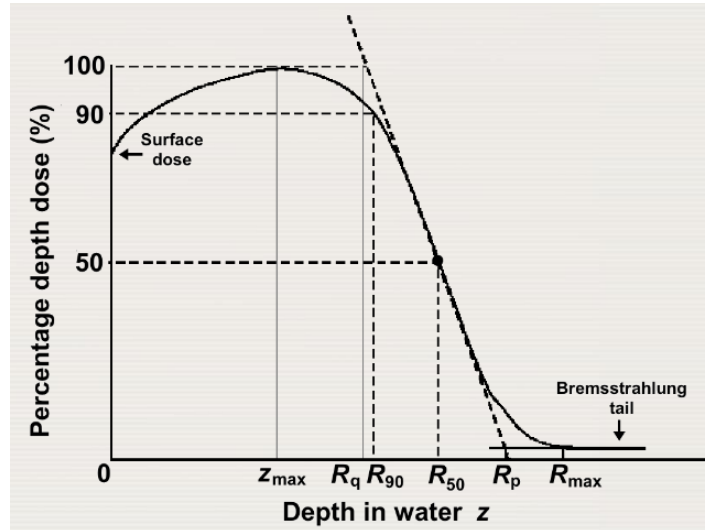
čia $\frac{d\sigma_{MCS}^2}{dZ'}$ yra kampinė sklaidos geba įvertinta vidutinei elektronų energijai T . Remiantis ICRU duomenimis (ICRU, 1972) vidutinė elektronų energija gylyje Z' apskaičiuojama

$$T(Z') = T_0 \left(1 - \frac{Z_{eff}(Z')}{R_p} \right), \quad (1.10)$$

čia T_0 – pradinė elektronų spindulio energija, Z_{eff} – efektinis gylis apskaičiuojamas

$$Z_{eff}(Z) = \int_{-L_0}^Z \frac{dE/dZ'}{(dE/dZ')_{H_2O}} dZ', \quad (1.11)$$

čia dE/dZ' – medžiagos stabdomoji geba, $(dE/dZ')_{H_2O}$ – vandens stabdomoji geba, R_p – praktinis elektronų diapozonas, kuris yra lygus gyliui, kuriame liestinė, išvesta per stačiausią elektrono gylio-dozės kreivės vietą, kerta horizontaliąją ašį, ši priklausomybė pavaizduota 10 paveiksle.



10 pav. Elektronų gylio-dozės kreivė pavaizduota kartu su praktiniu diapozonu (Strydom *et al.* 2012)

Elektronų spindulys palikęs tiesinio dalelių greitintuvo galvutę keliauja erdve ir skleidžiasi, todėl kiekvienas pieštukinis spindulys privalo turėti pradinį kampinį pasiskirstymą $\sigma_{\theta_x}^2 = \sigma_{\theta_y}^2$. Pastarasis pasiskirstymas projektuojamas į horizontalųjį pasiskirstymą, kurio reikšmė lygi $(Z + L_0)\sigma_{\theta_x}$. Gautos horizontaliųjų pasiskirstymų reikšmės sudedamos ir įstatomos į (1.8) formulę ir gaunama nauja (1.12) formulė.

$$f(X, Y, Z) = \frac{1}{2\pi\sigma^2} e^{-\frac{X^2 + Y^2}{2\sigma^2}}, \quad (1.12)$$

čia $\sigma^2 = (Z + L_0)^2 \sigma_{\theta_x}^2 + \sigma_{MCS}^2$. K. Hogstromas rekomenduoja σ_{θ_x} nustatyti atliekant spindulio pusšėsilio matavimus: sudaryti tiesinę funkciją $k(x)$, kuri aproksimuotų pusšėsilio ilgio ir gylio žemiau kolimatoriaus duomenis, tuomet funkcijos krypties koeficientą padauginti iš 0.391: $\sigma_{\theta_x} = 0.391k'(x)$. (Hogstrom *et al.* 1981).

Antroji (1.8) funkcijos komponentė $g(Z)$ aprašoma (1.13) išraiška:

$$g(Z) = g_0(Z_{eff}) \left(\frac{SSD + Z_{eff}}{SSD + Z} \right)^2. \quad (1.13)$$

Šios funkcijos kvadratinis narys įvertina atvirkštinio kvadrato dėsnį ir efektinio gylio Z_{eff} paklaidas. Čia SSD – atstumas tarp elektronų šaltinio ir kūno paviršiaus (angl. *source-skin distance*), g_0 – gylio-dozės funkcija vandenyje, kuri gali būti gauta iš D_0 – išmatuotų

gylio-dozės kreivių vandenyje, remiantis (1.12) ir (1.13) formulėmis:

$$D_0(X, Y, Z) = g_0(Z) \frac{1}{2\pi\sigma_0^2} \iint_Z e^{-\frac{(X-X')^2 + (Y-Y')^2}{2\sigma_0^2}} dX' dY'. \quad (1.14)$$

Toliau lygybėms aprašyti bus naudojamos klaidos funkcijos: $erf(x) = \frac{1}{\sqrt{\pi}} \int_{-x}^x e^{-t^2} dt$, taip pat pažymimas nagrinėjamo lauko dydis X ir Y kryptimis, atitinkamai WX ir WY . Į (1.6) formulę įrašomos gautos išraiškos: (1.12) ir (1.13)

$$\begin{aligned} D(X, Y, Z) = & \frac{1}{4} \left(erf \left(\frac{WXZ/2 - X}{\sqrt{2}\sigma} \right) + erf \left(\frac{WXZ/2 + X}{\sqrt{2}\sigma} \right) \right) \times \\ & \left(erf \left(\frac{WYZ/2 - Y}{\sqrt{2}\sigma} \right) + erf \left(\frac{WYZ/2 + Y}{\sqrt{2}\sigma} \right) \right) \times \\ & g_0(Z_{eff}) \left(\frac{SSD + Z_{eff}}{SSD + Z} \right)^2, \end{aligned} \quad (1.15)$$

čia $WXZ = WX(1 + Z/SSD)$, $WYZ = WY(1 + Z/SSD)$. Naudojant išmatuotą centrinės ašies gylio-dozės kreivę $D_0(0, 0, Z)$ nustatomas g_0 , kai nagrinėjamo lauko dydis pažymėtas $WX\phi$ ir yra stačiakampio formos:

$$\begin{aligned} D(X, Y, Z) = & \frac{1}{4} \left(erf \left(\frac{WXZ/2 - X}{\sqrt{2}\sigma} \right) + erf \left(\frac{WXZ/2 + X}{\sqrt{2}\sigma} \right) \right) \times \\ & \left(erf \left(\frac{WYZ/2 - Y}{\sqrt{2}\sigma} \right) + erf \left(\frac{WYZ/2 + Y}{\sqrt{2}\sigma} \right) \right) \times \\ & D_0(0, 0, Z_{eff}) erf \left(\frac{WX\phi/2}{\sqrt{2}\sigma} \right)^{-2} \left(\frac{SSD + Z_{eff}}{SSD + Z} \right)^2. \end{aligned} \quad (1.16)$$

Kitos formos laukų dozei apskaičiuoti naudojama (1.17) formulė:

$$\begin{aligned} D(X, Y, Z) = & \frac{1}{2\pi\sigma^2} \iint_Z S(X', Y') e^{-\frac{(X-X')^2 + (Y-Y')^2}{2\sigma^2}} dX' dY' \times \\ & D_0(0, 0, Z_{eff}) erf \left(\frac{WX\phi/2}{\sqrt{2}\sigma} \right)^{-2} \left(\frac{SSD + Z_{eff}}{SSD + Z} \right)^2. \end{aligned} \quad (1.17)$$

Hogstromo algoritmas remiasi duomenimis, kurie gaunami iš nagrinėjamo kūno kompiuterinės tomografijos nuotraukos. Esminė informacija apie nagrinėjamą kūną yra audinių stabdomoji ir sklaidomoji geba, kuri aprašoma kaip funkcija nuo kompiuterinės tomografijos skaičiaus.

Kompiuterinės tomografijos nuotraukoje informacija apie kūną įprastai pateikiama stačiakampio tinklelio vokseliuose, tačiau Hogstromo algoritmas reikalauja, kad nuotraukos duomenys būtų pateikiami vėduoklės (angl. *fan lines*) tinklelio forma, tačiau kartu horizontalios tinklelio kreivės privalo išlikti statmenos centriniam spinduliui. Kompiuterinės tomografijos skaičius apskaičiuojamas tiesių susikirtimo taškuose interpoliuojant turimus duomenis. (Parker *et al.* 1979)

Efektinių gylių matrica Z_{eff} apskaičiuojama vėduoklės tinklelio taškams remiantis (1.11) formule. Skaitinės efektinių gylių reikšmės gali būti gautos taikant skaitinio integravimo formulę:

$$Z_{eff}^{i,j} = Z_{eff}(Z_i, X_j) = \sum_{k=1}^{i-1} \left(\frac{dE/dZ}{(dE/dZ)_{H_2O}} \right)_{k,j} \Delta Z + \frac{\Delta Z}{2} \left(\frac{dE/dZ}{(dE/dZ)_{H_2O}} \right)_{i,j}, \quad (1.18)$$

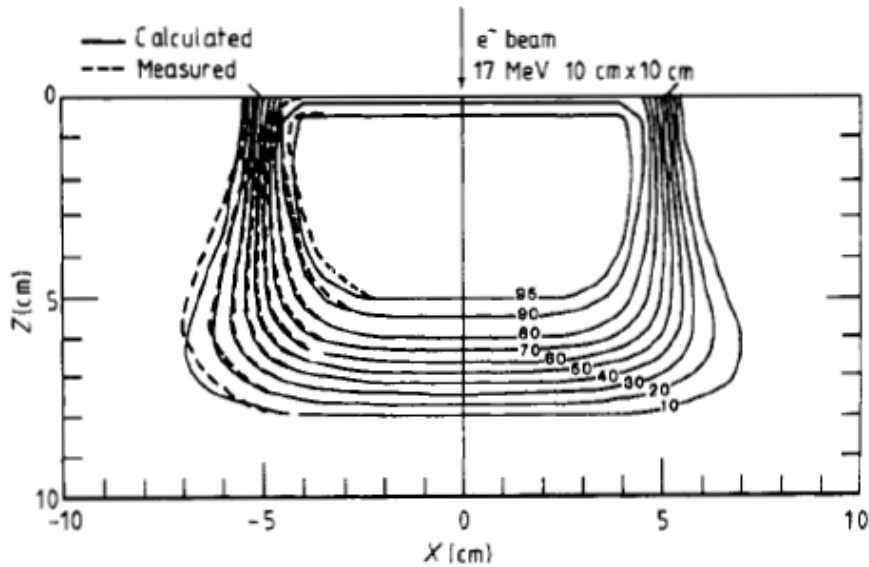
čia ΔZ – integravimo žingsnis pagal vėduoklės liniją. Matrica σ_{MCS}^2 taip pat skaičiuojama vėduoklės susikirtimo taškams remiantis (1.9) formule, skaitinėms reikšmės gauti – naudojama (1.19) formulė:

$$\begin{aligned} (\sigma_{MCS}^2)_{i,j} &= \sigma_{MCS}^2(Z_i, X_j) = \sum_{k=1}^{i-1} \left(\frac{d\sigma_{MCS}^2}{(dZ)} \right)_{k,j} \int_{Z_k}^{Z_k+\Delta Z} (Z_i - Z')^2 dZ' = \\ &= \sum_{k=1}^{i-1} \left[\left(\frac{d\sigma_{MCS}^2}{(dZ)} \right)_{k,j} \Delta Z \right] \left[(Z_k - Z_i)^2 + \Delta Z(Z_k - Z_i) + \frac{\Delta Z^3}{3} \right]. \end{aligned} \quad (1.19)$$

Skaičiavimams siūloma naudoti toliau pateikiamas formules:

$$\begin{aligned} (\sigma_{MCS}^2)_{i,j} &= [M_2^{i,j} + M_1^{i,j} + M_0^{i,j}/3]; \\ M_2^{i,j} &= M_2^{i-1,j} - 2M_1^{i-1,j} + M_0^{i-1,j} + \Delta Z(d\sigma_{MCS}^2/dZ)_{i-1,j}; \\ M_1^{i,j} &= M_1^{i-1,j} - M_0^{i-1,j} - \Delta Z(d\sigma_{MCS}^2/dZ)_{i-1,j}; \\ M_0^{i,j} &= M_0^{i-1,j} + \Delta Z(d\sigma_{MCS}^2/dZ)_{i-1,j}. \end{aligned} \quad (1.20)$$

Hogstromas pateikia fiziškai išmatuotų ir algoritmu apskaičiuotų rezultatų palyginimą vandenyje – 11 paveiksle pateiktas 17 MeV, elektronų spindulio dozės pasiskirstymas vandenyje, kai atstumas tarp spindulio šaltinio ir kūno paviršiaus yra 100 cm. Horizontali ir vertikali



11 pav. 17 MeV elektronų spindulio izodozinės linijos vandenyje (Hogstrom *et al.* 1980)

ašys žymi erdvės koordinatės išmatuotas centimetrais, punktyrine linija pažymėtos algoritmu apskaičiuotos izodozinės linijos, vientisa linija – fiziškai išmatuotos izodozinės linijos. Hogstromo straipsnyje (Hogstrom *et al.* 1981) pateikiama, kad tarp 10% ir 95% izodozinių linijų rezultatų skirtumas yra apie ± 3 mm, tarp likusių linijų – ± 1 mm.

Hogstromo algoritmas, kartu ir kiti, pieštukinio spindulio algoritmai turi trūkumų, dėl kurių apskaičiuotų rezultatų tikslumas gali nesutapti nagrinėjant skirtingus spinduliuotės ir kūno modelius. Pagrindiniai trūkumai aprašomi Kialo (Keall, 1996): (i) Eyges sklaidos teorija, kuria remiasi algoritmas yra riboto tikslumo ir neatsižvelgia į specifines spindulio sklaidos fizikines savybes, tokias kaip labai plataus kampo sklaida ar spindulio sklaida giliose kūno audiniuose; (ii) Hogstromo algoritmas yra nagrinėjamas $x - z$ plokštumoje, todėl daroma prielaida, kad y koordinatės atžvilgiu kūno struktūra nekinta. Dėl šios priežasties galimos paklaidos, kai nagrinėjame kūne aptinkamos nedidelės oro ar kaulų sritys; (iii) kiekvieno pieštukinio spindulio suteikiama dozė apskaičiuojama taip, lyg spindulys į kūno audinį kristų statmenu kampu, o audinio tipas nustatomas tik centrinio spindulio pozicijoje ir neatsižvelgiama į galimus pokyčius greta centrinio spindulio.

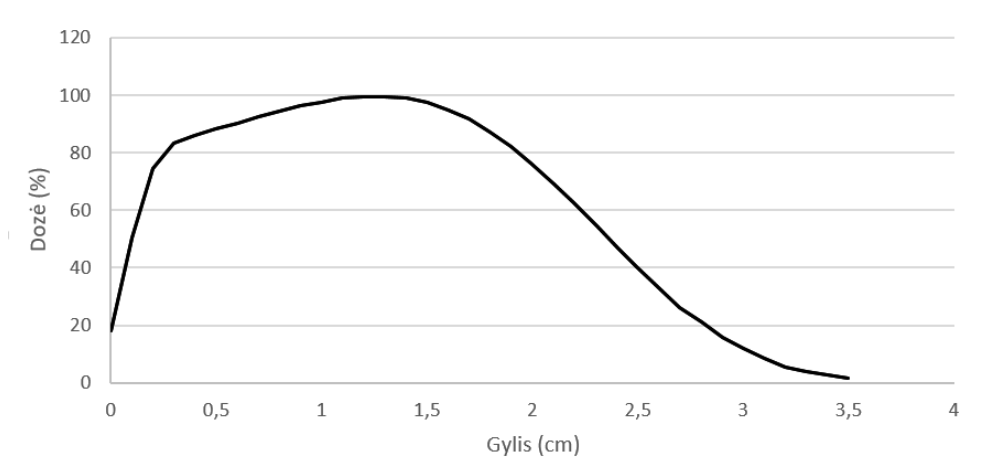
2. TIRIAMOJI DALIS

2.1. Hogstromo algoritmo pritaikymas

2.1.1. Duomenys

Šiame darbe naudojami duomenys gauti iš Lietuvos nacionalinio vėžio instituto: kompiuterinės tomografijos nuotraukos; 6 MeV elektronų spinduliuotės gylio-dozės vandenyje duomenys pavaizduoti 12 paveiksle; kompiuterinės tomografijos reikšmių skirtinguose audiniuose informacija; spinduliuotės pusšėšelio ilgio duomenys. Tiesinės sklaidomosios ir stabdomosios gebų santykis nagrinėjamoje medžiagoje su reikšme vandenyje nepriklauso nuo elektronų spinduliuotės energijos, kai ši energija kinta nuo 1 MeV iki 20 MeV. Remiantis ICRU (ICRU, 1972) duomenimis sudaroma 2 lentelė, kurioje pateikiamos santykinų sklaidomųjų ir stabdomųjų gebų reikšmės.

Kompiuterinės tomografijos nuotraukose saugoma informacija ne tik apie kūno struktūrą, tačiau kartu pateikiami duomenys apie kritinius organus t. y. tuos organus, kurių negalima pažeisti jonizuojančia spinduliuote, informacija apie spinduliuotės taikinį – naviką ir techninė nuotraukos informacija. 13 paveiksle pateikiamas pilnas, darbe nagrinėjamos kompiuterinės tomografijos nuotraukos vaizdas. Rausva spalva pažymėta sritis žymi piktybinį naviką, kurį



12 pav. 6 MeV energijos elektronų spinduliuotės gylio-dozės pasiskirstymas vandenyje

2 lentelė. Stabdomosios ir sklaidomosios gebų reikšmės skirtinguose audiniuose

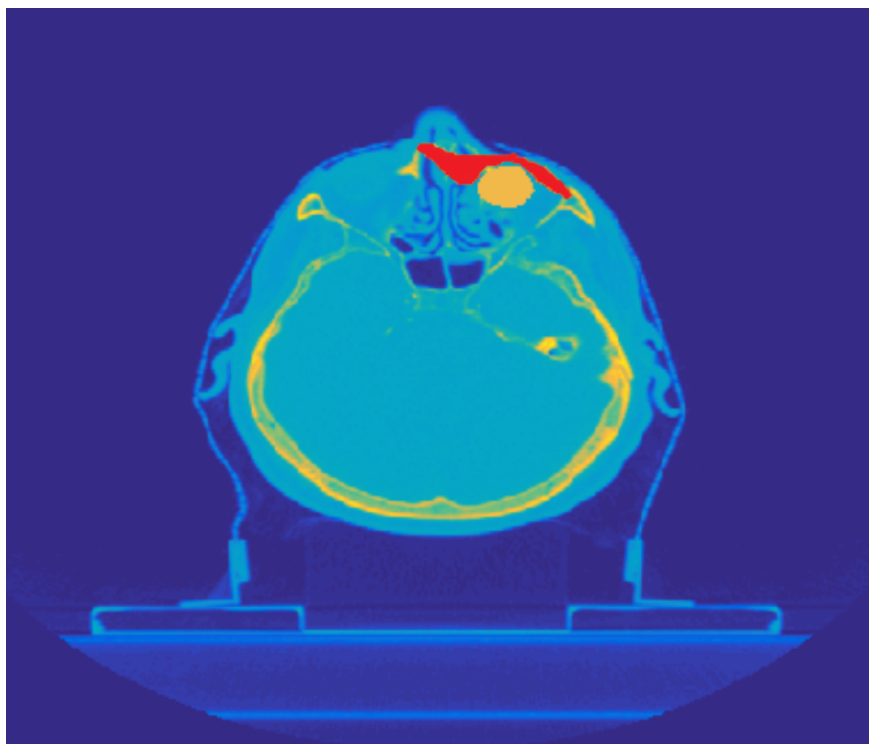
Audinys	HU	$\frac{dE/dZ}{(dE/dZ)_{H_2O}}$	$\frac{d\sigma^2/dZ}{(d\sigma^2/dZ)_{H_2O}}$
Plaučiai	-706	0,311	0,292
Riebalai	-102	0,933	0,729
Kaulų čiulpai	6	1,027	0,912
Smegenys	30	1,072	1,002
Inkstai	44	1,043	1,025
Raumenys	56	1,051	1,040
Kepenys	64	1,059	1,045
Kaulai	872	1,422	1,863

būtina paveikti spinduliuote; geltona spalva – kritinį organą, šiuo atveju paciento akį; aplink kūną matomas, šviesiai mėlynos spalvos kontūras – prietaisas naudotas kūno judesiams apriboti.

2.1.2. Vėduoklės linijų tinklelis

Hogstromo algoritmas yra apibrėžtas naudoti ant vėduoklės linijų tinklelio (angl. *fan line grid*), šis tinklelis sudaromas remiantis Parkerio (Parker *et al.* 1979) apibrėžtomis taisyklėmis:

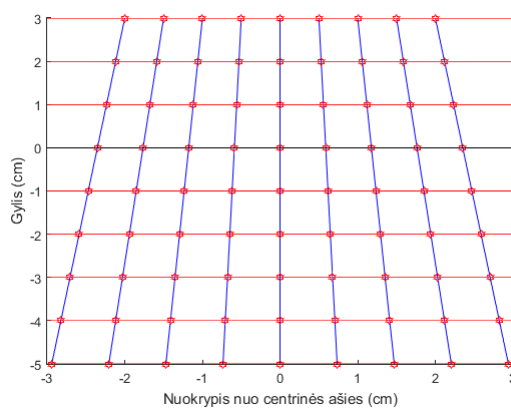
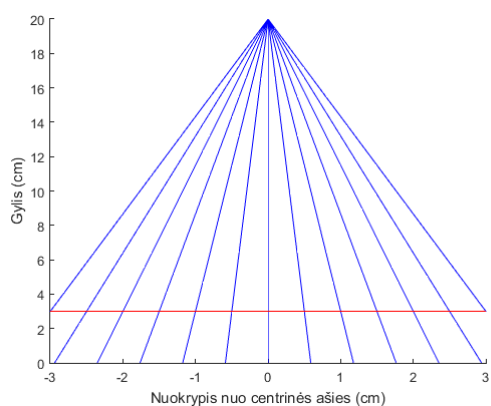
1. Iš elektronų spinduliuotės šaltinio brėžiami spinduliai nukreipti link nagrinėjamo kūno;
2. Antrinio kolimatoriaus aukštje atstumai tarp visų gretimų spindulių yra vienodi;
3. Pradedant nuo antrinio kolimatoriaus aukščio, brėžiamos vienodais atstumais nutolu-



13 pav. Paciento galvos kompiuterinės tomografijos nuotrauka

sios tiesės statmenos centrinei spinduliuotės ašiai;

4. Nustatomi spindulių ir tiesių susikirtimo taškai, kuriems bus priskirti kompiuterinės tomografijos skaičiai.



- (a) Spinduliuotė iš elektronų šaltinio nukreipta į (b) Vėduoklės linijų tinklėlis žemiau antrinio kolimatoriaus
nagrinėjamą kūną; raudona spalva pažymėta limatoriaus
tiesės žymi antrinio kolimatoriaus aukštį

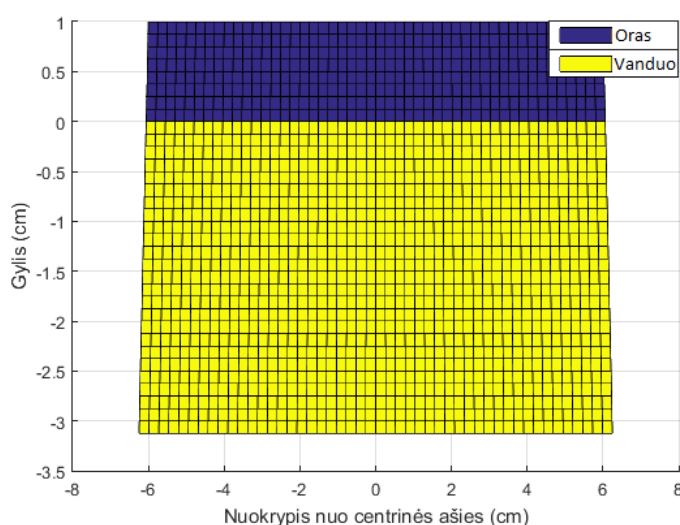
14 pav. Vėduoklės linijų tinklėlis

14 (b) paveiksle pavaizduotas vėduoklės linijų tinklėlis, kurio taškams priskiriami kompiuterinės tomografijos skaičiai. Skaičiai priskiriami juos interpoliuojant; interpoliuojama

remiantis duomenimis iš kompiuterinės tomografijos nuotraukos. Šiame paveiksle pateikiamas tik vaizdas pavyzdys – vertikalaus ir horizontalaus tinklelio žingsnių dydžiai įprastai yra mažesni nei pusė milimetro.

2.1.3. Spinduliuotės dozės pasiskirstymas vandenyje

Algoritmų ar metodų testavimui radioterapijos srityje naudojami vandens fantomai, tiriami kaip spinduliuotės dozė pasiskirsto vandenyje. Šios terpės savybės plačiai aprašomos įvairiuose darbuose, informaciją apie dozės matavimo standartus galima rasti oficialiuose tarptautinės atominės energijos agentūros leidiniuose (IAEA, 2000).



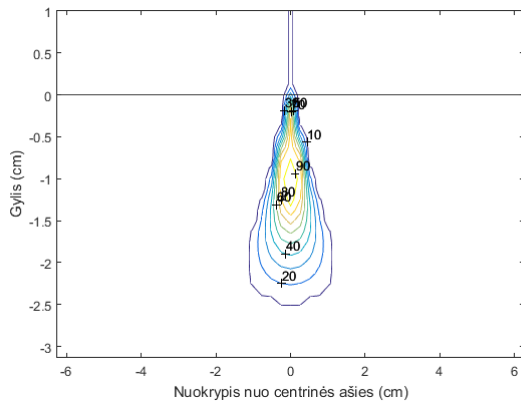
15 pav. Vandens fantomas

15 paveiksle pateikiamas naudojamas vandens fantomas: antrinis kolimatorius yra 1 cm aukštyje nuo nagrinėjamo kūno paviršiaus, šiuo atveju – vandens; mėlyna spalva žymima sritis yra oras, geltona – vanduo. Šis fantomas paveikiamas 6 MeV elektronų spinduliuote, 16 paveiksle pavaizduota fantomui suteiktos dozės kiekis, kai naudojamas vienas pieštukinis spindulys (a) ir pieštukinių spindulių srautas (b).

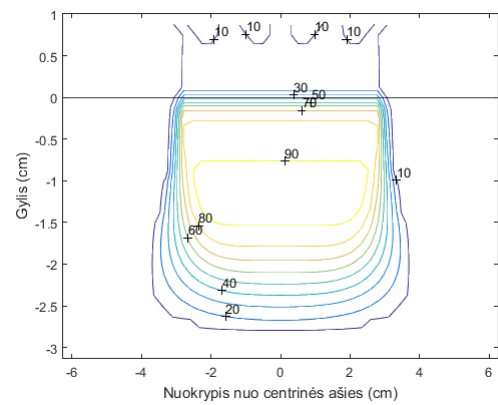
16 paveiksle matoma, kad 6 MeV elektronų spindulys vandens fantome maksimalią dozę pasiekia gylyje tarp 0,75 cm ir 1,5 cm. Pasiekus 90 % dozę, spinduliuotės dozė sumažėja iki 10 % po apytiksliai 1,2 cm, palyginti su 17 MeV spinduliuote – dozė nuo 90 % iki 10 % sumažėja po apytiksliai 2 cm, kaip matoma 11 paveiksle.

2.1.4. Spinduliuotės dozės pasiskirstymas paciento kūne

Elektronų spinduliuotės pasiskirstymui paciento kūne įtakos turi unikali geometrija ir įvairūs audiniai, kurių stabdomosios ir sklaidomosios savybės elektronų spinduliuotės atžvil-



(a) Pieštukinio spindulio dozės izolinijos vandenyje

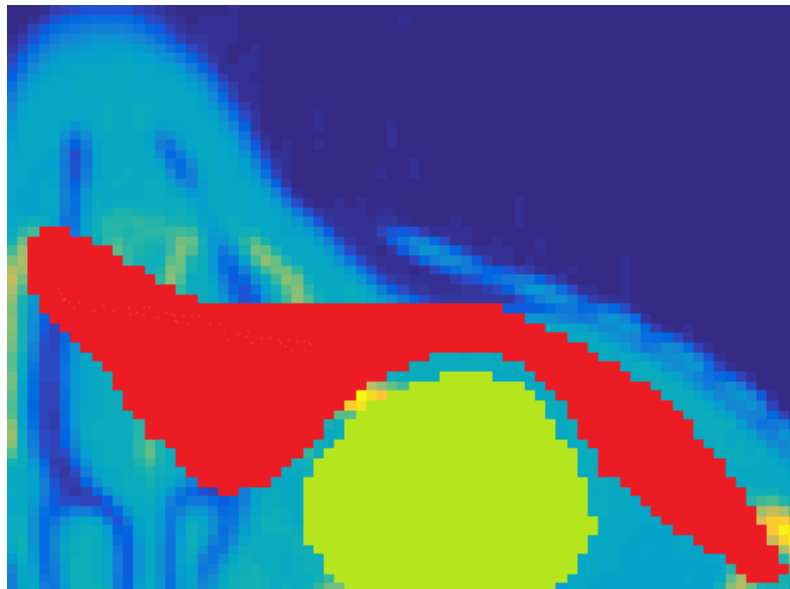


(b) Pieštukinių spindulių srauto dozės izolinijos vandenyje

16 pav. Elektronų spinduliuotės dozė vandens fantome

giu skiriasi. Dėl šios priežasties spinduliuotės suteikiama dozė dažnai tampa pasiskirsčiusi netolygiai, tačiau Hogstromo algoritmas atsižvelgia ir prisitaiko prie šių pokyčių.

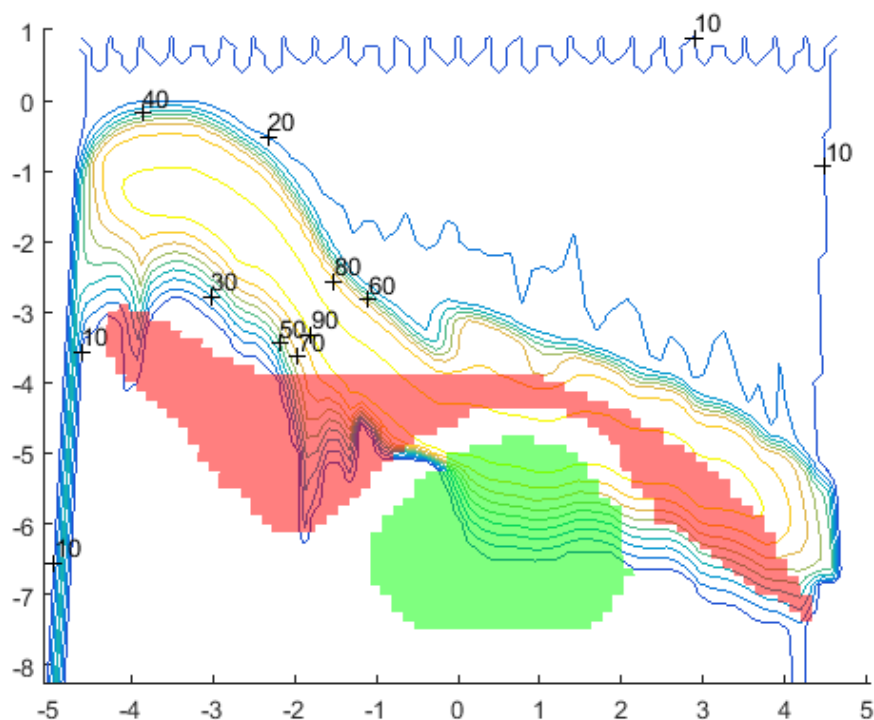
Iš 13 paveiksle pateiktos kompiuterinės tomografijos nuotraukos išskiriama sritis tolesniam nagrinėjimui, kuri pateikiama 17 paveiksle: raudona spalva pažymėtas piktybinis navikas; žalia spalva pažymėtas kritinis organas.



17 pav. Nagrinėjama kompiuterinės tomografijos nuotraukos sritis

Paciento kūno sritis, matoma 17 paveiksle, paveikiama elektronų spinduliuote, kurios energija yra 6 MeV. Laikoma, kad antrinis kolimatorius yra 1 cm aukštyje, atstumas tarp spinduliuotės šaltinio ir kūno – 100 cm, spinduliuotė į kūną nukreipta vertikalčiai žemyn. Spinduliuotės poveikio rezultatas matomas 18 paveiksle, kuriame pateikiamos suteiktos dozės

izolinijos kartu su piktybiniu naviku ir kritiniu organu.



18 pav. 6 MeV elektronų srauto dozės pasiskirstymas paciento kūne

18 paveiksle matoma, kad toks spinduliuotės pasiskirstymas kūne nėra priimtinas: dalis spinduliuotės, kurios dozė siekia 90 %, tenka kritiniui organui, ši spinduliuotės sritis bus valdoma naudojant boliusą; naudojama spinduliuotė nepasiekia visos piktybinio naviko srities – dalis lieka paveikta nepilnai arba nepaveikta visai, šiai problemai spręsti turėtų būti panaudotas papildomas spindulys, šiuo atveju būtų nukreiptas iš paveiklo viršutinės dešinėsios dalies į apatinių kairiąją arba turėtų būti naudojamas didesnės energijos elektronų spindulys.

2.2. Matematinis boliuso paviršiaus modelis

Optimalaus boliuso paviršiaus nustatymo uždavinys apibrėžiamas kaip daugiakriterio optimizavimo problema – siekiama suformuoti tokį boliuso paviršių, kuris apsaugotų kritinius kūno organus nuo žalingos jonizuojančios spinduliuotės, piktybiniam navikui tenkantis spinduliuotės dozės kiekis būtų maksimalus ir tolygiai pasiskirstęs, o likusiems audiniams tenkanti dozė turėtų būti kaip įmanoma mažesnė. Priklausomai nuo piktybinio naviko ir kritinio organo rūšies, specialistai numato slenkstines dozės ribas, kuriomis kiekvienas organas turėtų būti paveiktas: D_{Target}^{Min} – minimali dozės riba piktybiniam navikui; $D_{Critical}^{Max}$ –

maksimali dozės riba kritiniam organui; D_{Other}^{Max} – maksimali dozės riba kietiems audiniams.

Tokio tipo uždaviniai sprendžiami įvairiose mokslo srityse, pavyzdžiui, siekiant maksimalios naudos minimaliomis sąnaudomis (Hwang *et al.*, 2000), logistikos uždaviniuose (Trisna *et al.* 2016) ir kt. Tokiuose uždaviniuose, optimizavimo kriterijų gali būti nuo 2 iki k ir bendru atveju neegzistuoja vienas sprendinys, kuris būtų optimalus visų kriterijų atžvilgiu. Šiais atvejais sakoma, kad uždavinio tikslo funkcijos yra nesuderinamos ir egzistuoja Pareto optimalūs sprendiniai – turimą uždavinio sprendinį vieno kriterijaus atžvilgiu galima pagerinti tik kitų kriterijų sąskaita. (Miettinen, 1999). Bendru atveju daugiakriterio optimizavimo uždavinys gali būti apibrėžiamas kaip (2.1) išraiška.

$$\min_{x \in X} (f_1(x), f_2(x), \dots, f_k(x)) \quad (2.1)$$

čia $k \geq 2$ – optimizavimo kriterijų skaičius, X – galimų sprendinių aibė. Tarkime, kad $x_1, x_2 \in X$, sakoma, kad x_1 dominuoja x_2 , jei:

1. $f_i(x_1) \leq f_i(x_2) \forall i \in \{1, 2, \dots, k\}$;
2. $f_j(x_1) < f_j(x_2) \exists j \in \{1, 2, \dots, k\}$.

Sprendinys x^* laikomas Pareto optimaliu, jei jo nedominuoja joks kitas sprendinys iš galimų sprendinių aibės X . Pareto optimalių sprendinių aibė vadinama Pareto frontu (Hwang *et al.*, 2000).

Pagrindinis daugiakriterio optimizavimo uždavinių tikslas – rasti Pareto optimalius sprendinius. Kai nagrinėjama tikslo funkcija yra tolydi – Pareto optimalių sprendinių skaičius gali būti begalinis, tačiau dažnai šių sprendinių aibę rasti analitiškai negalima, todėl siekiama rasti tik tuos sprendinius, kurie kuo tolygiau padengtų Pareto frontą (Filatovas, 2012). E. Filatovas pateikia daugiakriterio optimizavimo metodų klasifikaciją:

- nepirminybės metodai (angl. *no preference*);
- aprioriniai metodai;
- aposterioriniai metodai;
- interaktyvūs metodai (angl. *interactive*).

Šiame darbe taikomi nepirminybės ir aposteriorinis metodai. Norint taikyti nepirminybės metodus, iš anksto turi būti žinoma metrika ir atramos taškai (angl. *reference point*) su

kuriais bus lyginamas sprendinys. Šiame darbe pasirinkta Euklido metrika, taikomi atramos taškai: D_{Target}^{Min} , $D_{Critical}^{Max}$, D_{Other}^{Max} . Remiantis šiuo metodu sudaryta (2.2) tikslo funkcija.

$$f(x, z) = \frac{\sum_{\forall(x,z) \in \text{Target}} \sqrt{(D_{Target}^{Min} - D_{Target}(x, z))^2}}{|N_{Target}|} + \frac{\sum_{\forall(x,z) \in \text{Critical}} \sqrt{(D_{Critical}^{Max} - D_{Critical}(x, z))^2}}{|N_{Critical}|} + \frac{\sum_{\forall(x,z) \in \text{Other}} \sqrt{(D_{Other}^{Max} - D_{Other}(x, z))^2}}{|N_{Other}|} \quad (2.2)$$

Šiame darbe taikomas aposteriorinis metodas, kuris transformuoja nagrinėjamą uždavinį į vienakriterį uždavinį naudojant skaliarizavimą (angl. *scalarizing*), tiksliau, taikomas svertinės kriterijų sumos metodas (angl. *weighted sum method*), kuris kiekvienai tikslo funkcijai suteikia svorį ω_i : $0 < \omega_i < 1$, $\sum_{i=1}^k = 1$ ir minimizuojama tikslo funkcijų suma. Šiuo metodu gauti sprendiniai, kai svoriniai koeficientai kinta tolygiai, nebūtinai bus pasiskirstę tolygiai ant Pareto fronto (Ritzel et al. 1994). Remiantis šiuo metodu sudaryta (2.3) tikslo funkcija.

$$g(x, z) = \omega_0 \frac{\sum_{\forall(x,z) \in \text{Target}} D_{Target}(x, z)}{|N_{Target}|} + \omega_1 \frac{\sum_{\forall(x,z) \in \text{Critical}} D_{Critical}(x, z)}{|N_{Critical}|} + \omega_1 \frac{\sum_{\forall(x,z) \in \text{Other}} D_{Other}(x, z)}{|N_{Other}|}; \quad (2.3)$$

čia ω_0 , ω_1 , ω_2 – svoriniai koeficientai; $|N_{Target}|$, $|N_{Critical}|$, $|N_{Other}|$ – taikinio, kritinių audinių ir kitų audinių pikselių skaičius kompiuterinės tomografijos nuotraukoje; $D(x, y)$ – procentinė dozės reikšmė nagrinėjamame taške. Aprašytoms (2.2) ir (2.3) funkcijoms optimizuoti toliau naudojami globalaus optimizavimo metodai: genetinis ir atkaitinimo (angl. *simulated annealing*) algoritmai.

2.2.1. Atkaitinimo algoritmas

Atkaitinimo algoritmas priklauso stochastinių algoritmų klasei ir dažnai naudojamas globalaus optimizavimo uždaviniams spręsti. Algoritmo pavadinimas kilo nuo analogijos metalurgijoje, kai įkaitintas metalas vėsdamas ir stingdamas įgauna kristalines, minimalios energijos struktūras – optimumą. Šis metodas gali būti efektyviai naudojamas, kai tikslo funkcija priklauso nuo daugelio kintamųjų. (Dzemyda et al. 2007)

Algoritmas susideda iš dviejų ciklų. Išorinis ciklas imituoja pradinės temperatūros mažėjimą – mažėjant temperatūrai mažėja ir aplinka, kurioje ieškoma naujo sprendinio, todėl paieška tampa vis lokalesnė. Vidiniame cikle generuojamas kaimyninis sprendinys, jei tikslo funkcijos reikšmė su šiuo sprendiniu geresnė – jis pakeičia ankstesnį sprendinį. Su tikimybe, kuri mažėja temperatūrai vėstant, geriausias, iš vidiniame cikle surastų taškų, gali būti

keičiamas blogesniu, tokiu būdu siekiama iširti platesnę sprendinių aibę ir pasiekti globalų optimumą. (Zhan *et al.* 2016)

Algoritmas 1 Atkaitinimo algoritmas

```

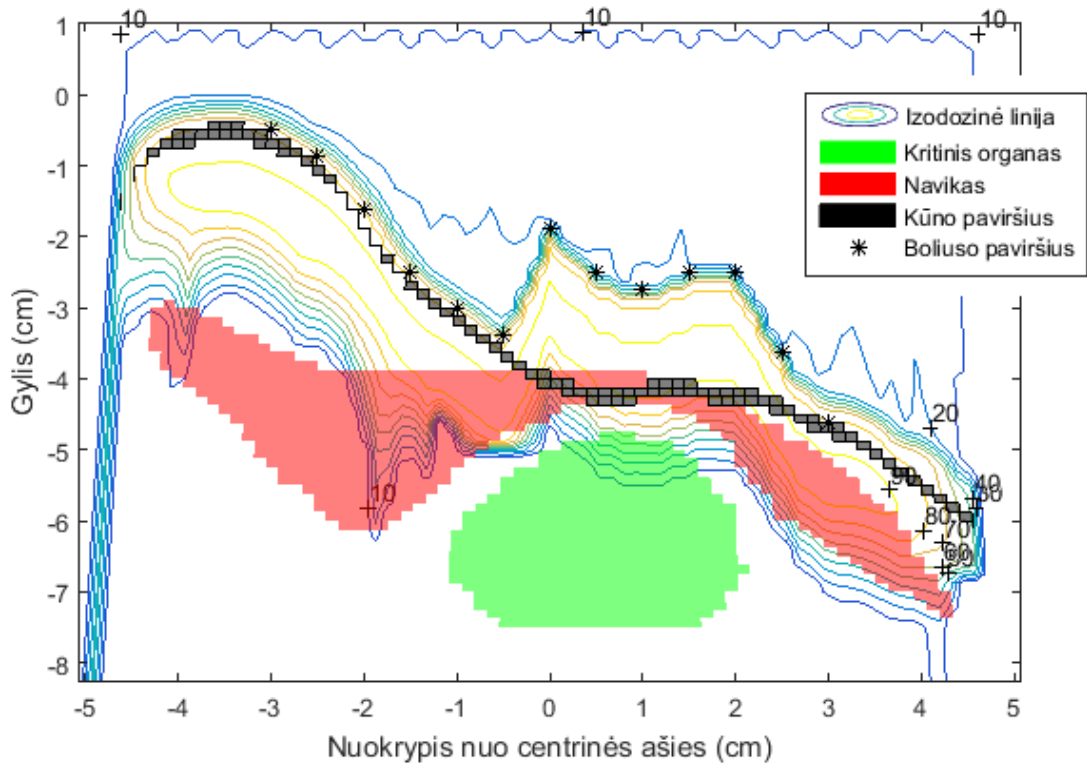
procedure SIMULATEDANALING(solution)
   $T \leftarrow 1$ ,  $T_{\min} \leftarrow 1e-6$ ,  $\alpha \leftarrow 0.9$ ,  $N \leftarrow 100$ ;
  cost  $\leftarrow$  cost(solution)
  while  $T > T_{\min}$  do
     $i \leftarrow 1$ 
    while  $i \leq N$  do
      newSolution  $\leftarrow$  neighbor(solution)
      newCost  $\leftarrow$  cost(newSolution)
      if acceptanceProb(cost, newCost, T)  $>$  random() then
        solution  $\leftarrow$  newSolution
        cost  $\leftarrow$  newCost
      end if
       $i \leftarrow i + 1$ 
    end while
     $T \leftarrow T * \alpha^n$ 
  end while
end procedure

```

Teisingas algoritmo parametrų T , α , N parinkimas yra ypatingai svarbus, tačiau iš anksto jų sužinoti nėra galimybės – kiekvieno uždavinio parametrai skiriasi, todėl dažnai parenkami eksperimentiniu būdu (Park *et al.* 1998).

Šiame darbe boliuso paviršiaus modelis pradedamas formuoti nuo odos paviršiaus ir yra nuosekliai formuojamas remiantis tikslo funkcijos rezultatais. Naudojama temperatūros vėsimos funkcija: $T = T\alpha^n$, kur n – iteracijos numeris vidiniame algoritmo cikle; tikimybė su kuria leidžiama priimti blogesnę sprendinį apskaičiuojama: $p = e^{-\Delta f/T}$, čia Δf – skirtumas tarp tikslo funkcijos reikšmių nagrinėjamuose taškuose, T – einamoji temperatūra. (Dzemyda *et al.* 2007).

Nepirminybės metodas. Bandymo metu, tikslo funkcijai (2.2) taikytos slenkstinės reikšmės: $D_{Critical}^{Max} = 0,1$; $D_{Targer}^{Min} = 0,8$; $D_{Other}^{Max} = 0,4$. Bandymo rezultatai pateikiami 19 paveiksle ir 3 lentelėje.



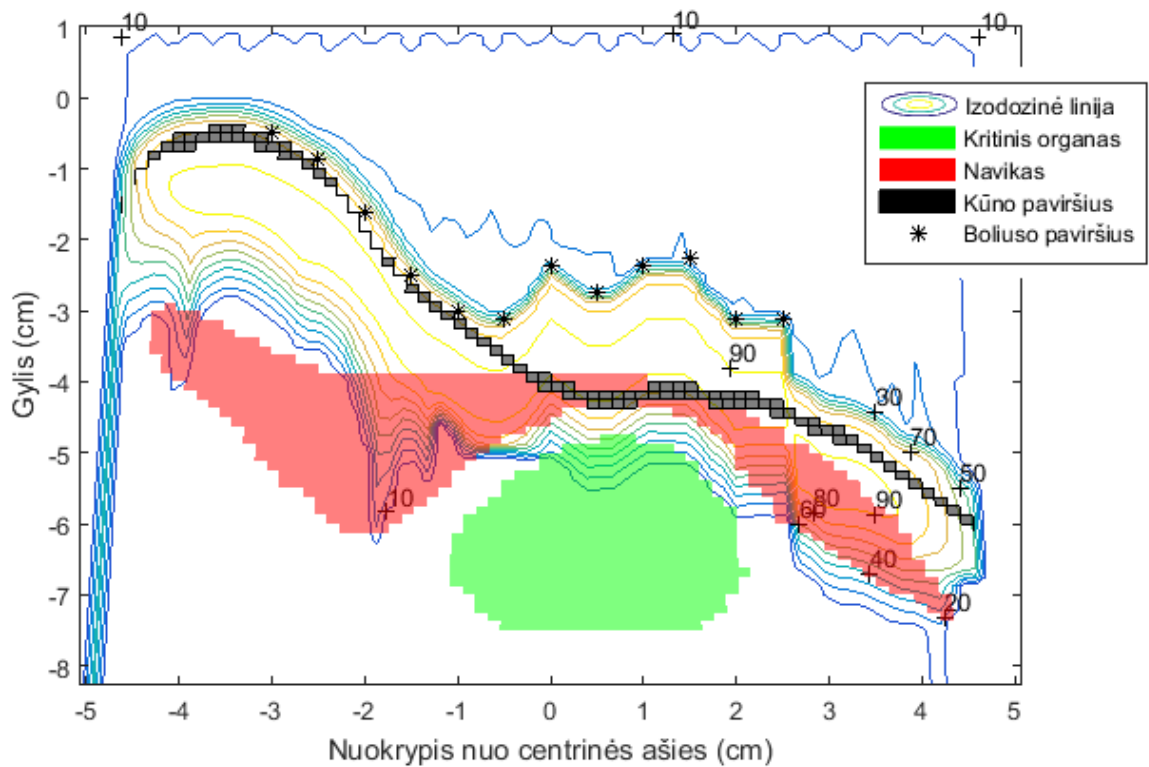
19 pav. Elektronų srauto dozės pasiskirstymas, kai boliuso formavimui naudotas atkaitinimo metodas ir (2.2) tikslo funkcija

3 lentelė. Boliuso poveikis dozės pasiskirstymui naudojant atkaitinimo metodą (1)

Audinys	Dozės vidurkis be boliuso (%)	Dozės vidurkis su boliusu (%)	Pokytis
Kritinis organas	25,155	1,935	-23,220
Navikas	43,875	41,484	-2,391
Kita	22,847	24,321	1,474

Bandymo metu gautas boliuso paviršius, kuris apsaugo kritinį organą nuo jonizuojančiosios spinduliuotės sumažindamas jam tenkanti dozės kiekį nuo 25,155 % iki 1,935 %, tačiau boliusas apsaugo ir piktybinį naviką sumažindamas jam tenkančią dozę nuo 43,875 % iki 41,484 %, taip pat kitiems audiniams tenkanti dozė padidėjo nuo 22,847 % iki 24,321 %. Šie rezultatai buvo pasiekti taikant parametrus: $T_0 = 100$, $T_{min} = 0,01$, $\alpha = 0,95$, $N = 50$.

Svertinės kriterijų sumos metodas. Bandymo metu naudota (2.3) tikslo funkcija su svertiniais koeficientais $\omega_0 = 0,4$; $\omega_1 = 0,5$; $\omega_2 = 0,1$. Gautas boliuso paviršius ir dozės pasiskirstymas pateikiamas 20 paveiksle ir 4 lentelėje.



20 pav. Elektronų srauto dozės pasiskirstymas, kai boliuso formavimui naudotas atkaitinimo metodas ir (2.3) tikslo funkcija

4 lentelė. Boliuso poveikis dozės pasiskirstymui naudojant atkaitinimo metodą (2)

Audinys	Dozės vidurkis be boliuso (%)	Dozės vidurkis su boliusu (%)	Pokytis
Kritinis organas	25,155	2,210	-22,945
Navikas	43,875	39,325	-4,550
Kita	22,847	24,463	1,616

Bandymo metu gautas boliuso paviršius, kuris apsaugo kritinį organą nuo jonizuojančiosios spinduliuotės sumažindamas jam tenkanti dozės kiekį nuo 25,155 % iki 3,538 %, tačiau boliusas apsaugo ir piktybinį naviką sumažindamas jam tenkančią dozę nuo 43,875 % iki 41,289 %, taip pat kitiems audiniams tenkanti dozė padidėjo nuo 22,847 % iki 24,309 %. Šie rezultatai buvo pasiekti taikant parametrus: $T_0 = 100$, $T_{min} = 0,01$, $\alpha = 0,95$, $N = 50$.

2.2.2. Genetinis algoritmas

Genetiniai algoritmai priskiriami metaeuristinių algoritimų klasei, dažnai naudojamų optimizavimo uždaviniams spręsti. Esminė algoritmo idėja remiasi principu „išlieka geriausiai

pritaikę“, kuris geriausiai žinomas kaip natūralios atrankos procesas gamtoje, todėl uždavinys įprastai aprašomas remiantis tokiais terminais kaip: populiacija, selekcija, mutacija, kryžminimas ir kt. Matematinio uždavinio kontekste – populiacija yra nagrinėjamų sprendinių aibė; selekcija, mutacija ir kryžminimas – procedūros skirtos populiacijai keistis, vystytis. (Faizan, 2017).

Bendru atveju genetiniai algoritmai neturi griežtai apibrėžtos realizacijos, kadangi nėra vieno „teisingo“ būdo, kuris tiktų visiems uždaviniams – G. Dzemyda (Dzemyda *et al.* 2007) pateikia ir aptaria galimas algoritmo realizacijas kaip pavyzdžiui, „valdomos atsitiktinės paieškos“. Bendras genetinio algoritmo pseudokodas pateikiamas 2 schemeje.

Genetiniai algoritmai negali garantuoti, kad rastas sprendinys bus globaliai optimalus. Rezultatams ypatingai daug įtakos turi pradinės sprendinių populiacijos parinkimas – sprendiniai privalo būti kuo įvairesni, kad nebūtų pasiektas tik lokalus optimumas. Genetinių algoritmų sudėtingumas sparčiai auga, kai didėja mutuoti galinčių uždavinio kintamųjų kiekis, todėl šios klasės algoritmai nėra rekomenduotini naudoti kompleksiniams uždaviniams spręsti kaip, pavyzdžiui, variklio ar statybų projekto modeliavimui, vietoje to, siūloma pasirinkti, pavyzdžiui, vieną variklio detalę ir optimizuoti (Sharma *et al.* 2015). Šiame darbe modeliuojamas boliuso paviršius, kaip nepriklausomas objektas, todėl genetiniai algoritmai šiam uždaviniui yra tinkami.

Toliau detalizuojami pagrindiniai šiame darbe naudojamo genetinio algoritmo etapai:

- inicializavimas – generuojama boliuso modelių aibė, kurių parametrai parenkami atsitiktinai, siekiant apimti kuo platesnę modelių įvairovę;
- selekcija – iš boliuso modelių aibės atrenkami pageidaujami modeliai. Siekiant išvengti elitizmo – tarp atrinktų modelių įtraukiami ir tie, kurių tinkamumas nėra tarp geriausių;
- kryžminimas – kiekvienam naujam vaikinui elementui sudaryti naudojami selekcijos etapo metu atrinkti populiacijos nariai: vaikinis narys perima dviejų tėvų savybes – pirmąją boliuso pusę gaunama iš pirmojo tėvo, antroji pusė – iš antrojo;
- mutacija – kiekvienas vaikinis elementas su numatyta tikimybe, gali įgauti naują savybę kurios neturėjo nei vienas iš tėvinių narių.

Nepirmenybės metodas. Bandymo metu, tikslo funkcijai (2.2) taikomos slenkstinės reikšmės: $D_{Critical}^{Max} = 0,1$; $D_{Targer}^{Min} = 0,8$; $D_{Other}^{Max} = 0,4$. Rezultatai pateikiami 19 paveiksle ir 5 lentelėje.

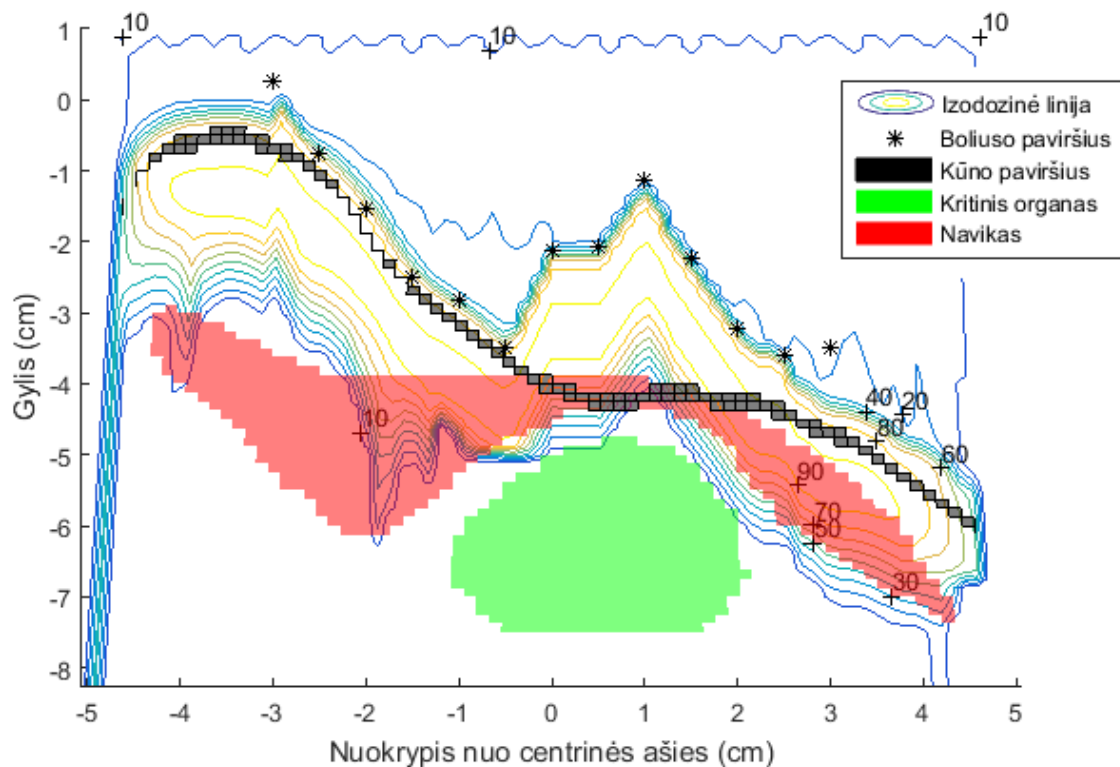
Algoritmas 2 Genetinis algoritmas

```
1: procedure GENETICALGORITHM
2:   Population  $\leftarrow$  InitializePopulation(Populationsize, Problemsize)
3:   EvaluatePopulation(Population)
4:    $S_{best} \leftarrow$  getSolution(Population)
5:   while StopCondition() do
6:     Parents  $\leftarrow$  SelectParents(Population, Populationsize)
7:     Children  $\leftarrow \emptyset$ 
8:     for Parent1, Parent2  $\in$  Parents do
9:       Child1, Child2  $\leftarrow$  crossOver(Parent1, Parent2)
10:      Children  $\leftarrow$  Mutate(Child1)
11:      Children  $\leftarrow$  Mutate(Child2)
12:    end for
13:    EvaluatePopulation(Children)
14:     $S_{best} \leftarrow$  getSolution(Children)
15:  end while
16: end procedure
```

5 lentelė. Boliuso poveikis dozės pasiskirstymui naudojant genetinį algoritmą (1)

Audinys	Dozės vidurkis be boliuso (%)	Dozės vidurkis su boliusu (%)	Pokytis
Kritinis organas	25,155	0,732	-24,423
Navikas	43,875	41,484	-2,391
Kita	22,847	24,321	1,474

Bandymo metu gautas boliuso paviršius, kuris apsaugo kritinį organą nuo jonizuojančiosios spinduliuotės sumažindamas jam tenkanti dozės kiekį nuo 25,155 % iki 0,732 %, tačiau boliusas apsaugo ir piktybinį naviką sumažindamas jam tenkančią dozę nuo 43,875 % iki 41,484 %, taip pat kitiems audiniams tenkanti dozė padidėjo nuo 22,847 % iki 24,321 % . Šie rezultatai buvo pasiekti po 38 populiacijos kartų.



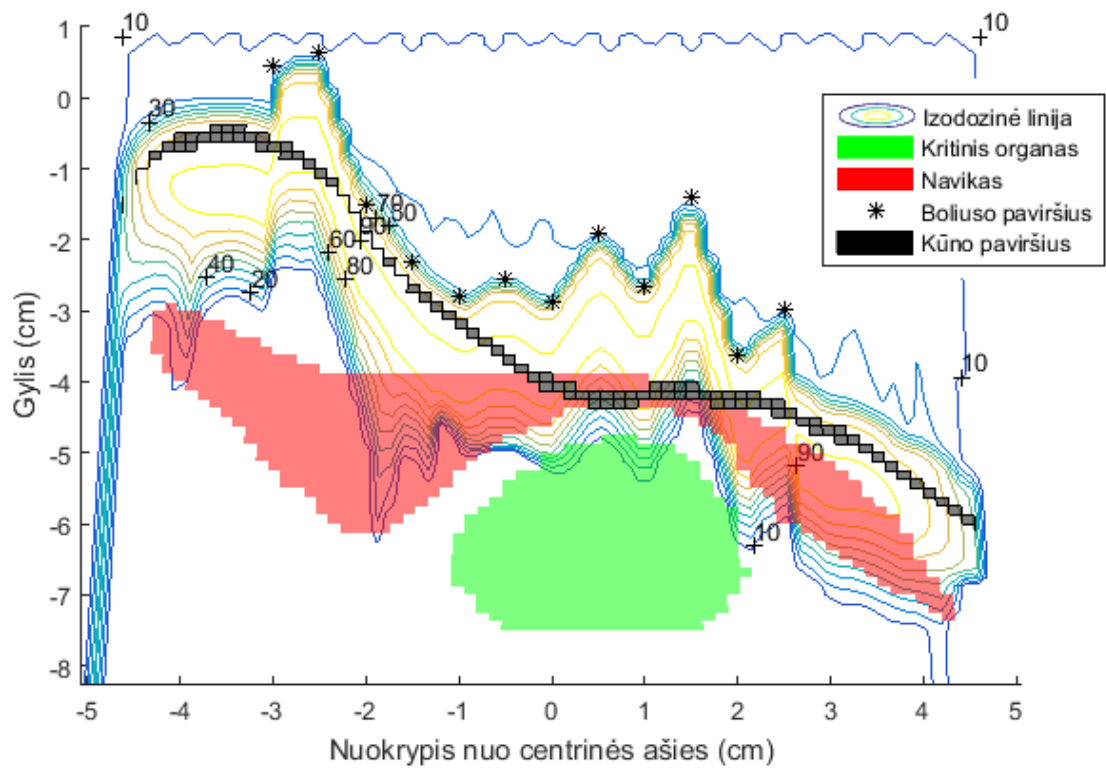
21 pav. Elektronų srauto dozės pasiskirstymas, kai boliuso formavimui naudotas genetinis algoritmas ir (2.2) tikslo funkcija

Svertinės kriterijų sumos metodas. Bandymo metu naudota (2.3) tikslo funkcija su svertiniais koeficientais $\omega_0 = 0,4$; $\omega_1 = 0,5$; $\omega_2 = 0,1$. Gautas boliuso paviršius ir dozės pasiskirstymas pateikiamas 22 paveiksle ir 6 lentelėje.

6 lentelė. Boliuso poveikis dozės pasiskirstymui naudojant genetinį algoritmą (2)

Audinys	Dozės vidurkis be boliuso (%)	Dozės vidurkis su boliusu (%)	Pokytis
Kritinis organas	25,155	2,210	-22.945
Navikas	43,875	39,325	-4.550
Kita	22,847	24,463	1.616

Bandymo metu gautas boliuso paviršius, kuris apsaugo kritinį organą nuo jonizuojančiosios spinduliuotės sumažindamas jam tenkanti dozės kiekį nuo 25,155 % iki 2,210 %, tačiau boliusas apsaugo ir piktybinį naviką sumažindamas jam tenkančią dozę nuo 43,875 % iki 39,325 %, taip pat kitiems audiniams tenkanti dozė padidėjo nuo 22,847 % iki 24,463 % . Šie rezultatai buvo pasiekti po 31 populiacijos kartos.



22 pav. Elektronų srauto dozės pasiskirstymas, kai boliuso formavimui naudotas genetinis algoritmas ir (2.3) tikslo funkcija

IŠVADOS

Baigiamajame darbe realizuotas analitinis Hogstromo algoritmas naudojant „MATLAB R2015a“ programinę įrangą. Apskaičiuotas 6 MeV elektronų srauto dozės pasiskirstymas vandenyje ir žmogaus kūne remiantis duomenimis iš kompiuterinės tomografijos nuotraukos. Sudarytas matematinis boliuso paviršiaus modelis sprendžiant daugiakriterio optimizavimo uždavinius genetiniu ir atkaitinimo metodais. Optimizavimo uždaviniams suformuluoti, buvo taikomi nepirminybės ir aposteriorinis metodai. Pirmajam metodui parinkti atramos taškai remiantis slenkstiniais, audiniams leidžiamos dozės reikalavimais, kurie randami medicinos fizikos žurnaluose.

Nepriklausomai nuo pasirinktos tikslo funkcijos ar optimizavimo metodo, gautas boliuso modelis pastebimai sumažino naudoto elektronų srauto dozės kiekį tenkantį kritiniam organui, nereikšmingai padidino dozės kiekį tenkantį kitiems kūno audiniams ir nereikšmingai sumažino dozės kiekį tenkantį spinduliuotės taikiniui. Skaičiavimo laiko atžvilgiu, geresni rezultatai gauti taikant atkaitinimo metodą. Norint efektyviai taikyti genetinį algoritmą, reikalinga didelė ir įvairi sprendinių populiacija, tačiau kiekvieno sprendinio įvertinimas yra laikui imli procedūra, todėl bendras algoritmo vykdymo laikas sparčiai auga.

Tęsiant tyrimus šioje srityje rekomenduojama atlikti lyginamąją analizę tarp Hogstromo algoritmu ir Monte Karlo metodais gautų rezultatų bei įvertinti kitų analitinių metodų panaudojimo galimybę elektronų srauto dozės pasiskirstymui apskaičiuoti. Atlikus lyginamąją analizę ir įvertinus analitinio metodo teisingumą – siūloma modeliuoti boliuso paviršių trimačiu atveju.

Literatūra

- [1] Almond, P. R.; Wright, A. E.; Boone, M. L. M. *High energy electron dose perturbations in regions of tissue heterogeneity*. 1967. Radiology. 7 p.
- [2] Dzemyda, G.; Šaltenis, V.; Tiešis, V. *Optimizavimo metodai*. 2007. Vilnius: Matematikos ir informatikos institutas. 166 p.
- [3] Tessier, F.; Mainegra, E. *EGSnrc: software tool to model radiation transport*. 2015. [interaktyvus], [žiūrėta 2018-02-23]. Prieiga per internetą: https://www.nrc-cnrc.gc.ca/eng/solutions/advisory/egsnrc_index.html.
- [4] Eyges, L. *Multiple Scattering with Energy Loss*. 1948. [interaktyvus], [žiūrėta 2017-11-25]. 10 p. Prieiga per internetą: <https://link.aps.org/doi/10.1103/PhysRev.74.1534>.
- [5] Eldib, A. A.; ElGohary, M. I.; Fan, J.; Jin, L.; Li, J.; Ma, C. C.; Elsherbini, N. A. *Dosimetric characteristics of an electron multileaf collimator for modulated electron radiation therapy*. 2009. Journal of applied clinical medical physics [interaktyvus], [žiūrėta 2018-04-03]. 18 p. Prieiga per internetą: <https://aapm.onlinelibrary.wiley.com/doi/pdf/10.1120/jacmp.v11i2.2913>.
- [6] Chamberland, E.; Beaulieu, L.; Lachance, B. *Evaluation of an electron Monte Carlo dose calculation algorithm for treatment planning*. 2015. Radiation Oncology Physics [interaktyvus], [žiūrėta 2018-04-04]. 21 p. Prieiga per internetą: <https://doi.org/10.1120/jacmp.v16i3.4636>.
- [7] Faizan, A. *Introduction to Genetic Algorithm (GA)*. 2017. Kansas [interaktyvus], [žiūrėta 2018-03-04]. 7 p. Prieiga per internetą: <http://engineerexperiences.com/introduction-to-genetic-algorithm-ga.html>.
- [8] Filatovas, E. *Daugiakriterių optimizavimo uždavinių sprendimas interaktyviuoju būdu*. 2012. Vilnius [interaktyvus], [žiūrėta 2018.04.01]. 139 p. Prieiga per internetą: http://old.mii.lt/files/mii_dis_2012_filatovas.pdf.
- [9] Goudsmit, S.; Sauderson, J. L. *Multiple scatter of electrons*. 1940. American Physical Society. 5 p.

- [10] Hogstrom, K. R.; Almond, P. R. *Review of electron beam therapy physics*. 2006. Physics in Medicine and Biology [interaktyvus], [žiūrėta 2018-03-18]. 35 p. Prieiga per internetą: [https://www.sprmn.pt/pdf/pmb6_13_r25_Electron_beam_therapy_\(Hogstrom-&Almond\).pdf](https://www.sprmn.pt/pdf/pmb6_13_r25_Electron_beam_therapy_(Hogstrom-&Almond).pdf).
- [11] Hogstrom, K. R.; Almond, P. R.; Mills, M. D. *Electron beam dose calculations*. 1981. Physics in Medicine and Biology [interaktyvus], [žiūrėta 2018-02-01]. 15 p. Prieiga per internetą: <https://www.ncbi.nlm.nih.gov/pubmed/6787621>.
- [12] Hogstrom, K.; Shiu, A. S. *Pencil-beam redefinition algorithm for electron dose distributions*. 1991. Medical Physics [interaktyvus], [žiūrėta 2018-04-06]. 11 p. Prieiga per internetą: <https://www.ncbi.nlm.nih.gov/pubmed/2008174>.
- [13] Hwang, C.; Masud, A. *Multiple objective decision making, methods and applications*. 1979. Springer-Verlag. 351 p.
- [14] International atomic energy agency *Absorbed dosedetermination in external beam radiotherapy*. 2000. Viena [interaktyvus], [žiūrėta 2018-04-28]. 242 p. Prieiga per internetą: https://www-pub.iaea.org/MTCD/publications/PDF/TRS398_scr.pdf.
- [15] ICRU *Radiation dosimetry: electron beams with energies between 1 and 50 MeV*. 1972. Merilandas. 168 p.
- [16] Keall, P. J. *The angular and energy distribution of a 15 MeV beam*. 1996. Adelaidė. 218 p.
- [17] Keall, P. J. *Electron Transport in Photon and Electron Beam Modelling*. 1994. Adelaidė. 42 p.
- [18] Khan, F. M.; Gibbons, J. P. *Khan's The Physics of Radiation Therapy*. 2003. Filadelfija. 565 p.
- [19] Kim, S. J.; Kim, D. H.; Kim, S. K. *Comparison of Pencil beam, Collapsed cone and Monte - Carlo algorithm in radiotherapy treatment planning for 6 MV photon*. 2015. Medical Physics [interaktyvus], [žiūrėta 2018-02-02]. 17 p. Prieiga per internetą: <https://arxiv.org/abs/1503.03224>
- [20] Mackie, T. R.; Battista, J. J. *A macroscopic Monte Carlo method for electron beam dose calculations: a proposal*. 1984. Use of Computers in Radiation Therapy. 4 p.

- [21] Miettinen, K. *Nonlinear Multiobjective Optimization*. 1999. Springer Science & Business Media. 298 p.
- [22] Nelson, W. R.; Hirayamal, H.; Rogers, D. W. *The EGS4 code system*. 1985. Kalifornija. 206 p.
- [23] Kooy, H. M.; Rashid, H. *A three-dimensional electron pencil beam algorithm*. 1989. Physics in Medicine and Biology. 14 p.
- [24] Park, M.; Kim, Y. *A systematic procedure for setting parameters in simulated annealing algorithms*. 1998. [interaktyvus], [žiūrėta 2018-04-02]. 11 p. Prieiga per internetą: <https://pdfs.semanticscholar.org/78d0/363647d5ecb68956c9ee3fab80005618b083.pdf>.
- [25] Parker, R. P.; Hobday, P. A.; Cassell, K. J. *The direct use of CT numbers in radiotherapy dosage calculations for inhomogeneous media*. 1979. Physics in Medicine and Biology [interaktyvus], [žiūrėta 2018-04-01]. 9 p. Prieiga per internetą: <https://www.ncbi.nlm.nih.gov/m/pubmed/472014/>.
- [26] Perry, D. J.; Holt, J. G. *A model for calculating the effects of small inhomogeneities on electron beam dose distributions*. 1980. Physics in Medicine and Biology [interaktyvus], [žiūrėta 2018-02-01]. 8 p. Prieiga per internetą: <https://aapm.onlinelibrary.wiley.com/doi/abs/10.1118/1.594687>.
- [27] Ritzel, B. J.; Wayland, E. J.; Ranjithan, S. *Using Genetic Algorithms to Solve a Multiple Objective Groundwater Pollution Containment Problem*. 1994. Water Resources Research. 14 p.
- [28] Romeijn, H. E.; Ahuja, R. K.; Dempsey, J. F.; Kumar, A. *Column generation approach to radiation therapy treatment planning using aperture modulation*. 2005. Society for Industrial and Applied Mathematics [interaktyvus], [žiūrėta 2018-02-01]. 25 p. Prieiga per internetą: <https://epubs.siam.org/doi/abs/10.1137/040606612?journalCode=sjope8>.
- [29] Sharma, Y.; Gupta, N.; Dayanand. *Genetic Algorithms-Overview, Limitations and Solutions*. 2015. Advances in Computer Science and Information Technology. 5 p.
- [30] Strydom, W.; Parker, W.; Olivares, M. *Electron beams: physical and clinical aspects*. 2012. International atomic energy agency [interaktyvus], [žiūrėta 2018-02-01]. 10 p.

- vus], [žiūrėta 2018-02-02]. 91 p. Prieiga per internetą: http://www-naweb.iaea.org/nahu/DMRP/documents/slides/Chapter_08_Electron_beams.pdf.
- [31] Trisna, T.; Marimin, M.; Arkeman, Y.; Sunarti, T. *Multi-objective optimization for supply chain management problem: A literature review*. 2016. Decision Science Letters. 35 p.
- [32] Zankowski, C.; Laitinen, M.; Neuenschwander, H. *Fast Electron Monte Carlo for Eclipse*. 2004. Palo Alto. 24 p.
- [33] Zhan, S.; Lin, J.; Zhang, Z.; Zhong, Y. *List-Based Simulated Annealing Algorithm for Traveling Salesman Problem*. 2016. Comput Intell Neurosci [interaktyvus], [žiūrėta 2018-05-24]. 12 p. Prieiga per internetą: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4808530/>.

PRIEDAI

```
1 %PROJECT_PATH = ''; %Define Project path
2 SCAN_DATA_FOLDER = 'ScanData\';
3 PTV_NAME = 'RS.PTV.mat';
4 OD_NAME = 'RS.OD.mat';
5 SCAN_NAME = 'CT';
6
7 %Read data
8 if ~exist('excelDataLoaded', 'var') || excelDataLoaded == false
9     fprintf('BEGIN: read excel data \n');
10    [xStopPower, yStopPower] = readExcel('Data\StoppingPowerRatio4
        ', 1, 'B2:C12');
11    [xScatterPower, yScatterPower] = readExcel('Data\
        ScatteringPowerH2O ',1, 'A2:C34');
12
13    [xProfile1cm, yProfile1cm] = readExcel('Data\Profiliai', 2, '
        A3:B52');
14    [xProfile2cm, yProfile2cm] = readExcel('Data\Profiliai', 2, '
        E3:F53');
15
16    [xDepthDose, yDepthDose] = readExcel('Data\GylioDoze', 1, 'A2:
        B37');
17
18    excelDataLoaded = true;
19    fprintf('END: read excel data \n');
20 else
21    fprintf('Excel data already loaded\n');
22 end
23
24 %CSAPI Cubic spline interpolant with not-a-knot end condition.
25 deapthDose = csapi(xDepthDose, yDepthDose);
26
```

```

27 length1Spline = csapi(xProfile1cm , yProfile1cm);
28 length2Spline = csapi(xProfile2cm , yProfile2cm);
29
30 %used for halfshadow length
31 length1cm = fnval(length1Spline , 0.1) - fnval(length1Spline , 0.9);
32 length2cm = fnval(length2Spline , 0.1) - fnval(length2Spline , 0.9);
33
34 stoppingPowerRatio = csapi(xStopPower, yStopPower);
35 scaterPowerH20 = csapi(xScatterPower , yScatterPower);
36 H2OScatterRatio = 1;
37
38 sliceNo=185;
39 fprintf('BEGIN: get scan images \n');
40 SCAN_FILE = strcat(SCAN_NAME, '. ', int2str(sliceNo) , '.dem');
41 [vezysFull , saugaFull , scanImageFull] = getScanImages(strcat(
    PROJECT_PATH,SCAN_DATA_FOLDER), PTV_NAME, OD_NAME, SCAN_FILE,
    sliceNo);
42 fprintf('END: get scan images \n');
43
44 fprintf('BEGIN: crop data \n');
45 [vezys , sauga , patientCT , xDomain, zDomain] = cropScanImages(
    vezysFull , saugaFull , scanImageFull );
46 fprintf('END: crop data \n');
47
48 fprintf('BEGIN: get CT Value \n');
49 [X, Z, vezysValue , saugaValue , CTValue] = getCTValue( xDomain,
    zDomain, vezys , sauga , patientCT );
50 fprintf('END: get CT Value \n');
51
52 [m, n] = size(X);
53 RatioValue = fnval(stoppingPowerRatio , CTValue);
54 z_effValue = z_effFunc( RatioValue , X, Z );
55

```

```

56 T_0 = 6;
57 R_p = 3.8;
58 sigmaValue = sigmaCalc( X, Z, z_effValue, RatioValue,
    scatterPowerH20, T_0, R_p );
59
60 center = 0;
61 tangentShadow = (length2cm-length1cm)/(2-1);
62 L0 = 1;
63 SSD = 100;
64 intSteps = 200;
65
66 %Integravimo rezis nurodomas funkcijoje
67 [probValue, sigma_medValue, integralValue] = ...
68     prob_sigmaMed_integralValues( sigmaValue, X, Z, L0,
    tangentShadow, center, intSteps);
69 %
70 [ValueDoseMeas, ValueZDose] = doseCalc( X, Z, L0, SSD,
    tangentShadow, deapthDose, sigmaValue, intSteps);
71
72 doseMeasSpline = csapi(ValueZDose, ValueDoseMeas);
73 measuredAreaDose = doseValueCalc(z_effValue, integralValue,
    doseMeasSpline, SSD);
74
75 critGoal = 10;
76 cancerGoal = 80;
77 otherGoal = 30;
78
79 [critOld, cancerOld, score] = getScores( saugaValue, vezysValue,
    ...
80     measuredAreaDose, cancerGoal, critGoal, otherGoal);
81
82 skinEdge = getSkinEdge(CTValue);
83

```

```

84 numberOfPoints = 12;
85 [bolusPoints, edgePoints] = getBolusPoints( X, Z, skinEdge,
      numberOfPoints);
86
87 GA_solution = GeneticOptimization( 10, X, Z, ratioValue,
      deapthDose, ...
88     H2OScatterRatio, scatterPowerH20, ...
89     skinEdge, edgePoints, A, numberOfPoints, dx, dz, L0, SSD, T_0,
      R_p, ...
90     tangentShadow, center, intSteps, ...
91     criticalValue, tumorValue, tumorGoal, criticalGoal, otherGoal)
      ;
92
93 SA_solution = SimulatedAnealing( X, Z, RatioValue, saugaValue,
      vezysValue, bolusPoints, numberOfPoints, edgePoints, ...
94     scaterPowerH20, tangentShadow, deapthDose, score, dx, dz,
      center, intSteps, ...
95     L0, T_0, R_p, SSD, ...
96     cancerGoal, critGoal, otherGoal);

1 function [ vezys, sauga, CTImage ] = getScanImages( path, PTVName,
      ODName, scanName, sliceNo )
2     PTV = load(strcat(path,PTVName));
3     vezys = double(rot90(PTV.contours.Segmentation(:, :, sliceNo),
      3));
4
5     OD = load(strcat(path,ODName));
6     sauga = double(rot90(OD.contours.Segmentation(:, :, sliceNo), 3)
      );
7
8     CTImageFresh = dicomread(strcat(path,scanName));
9     CTInfo = dicominfo(strcat(path,scanName));
10
11     CTImage = double(CTImageFresh)*CTInfo.RescaleSlope+CTInfo.

```

```

        RescaleIntercept;
12 end

1 function [ vezysCrop, saugaCrop, scanImageCrop, xDomain, zDomain ]
    = cropScanImages( vezysFull, saugaFull, scanImageFull )
2     centerZ=128;
3     centerX=286;
4     cropSizeZ=30;
5     cropSizeX=37;
6
7     pixelSize=0.125;
8
9     vezysCrop = vezysFull(centerZ - cropSizeZ : centerZ +
        cropSizeZ, centerX - cropSizeX : centerX + cropSizeX);
10    saugaCrop = saugaFull(centerZ - cropSizeZ : centerZ +
        cropSizeZ, centerX - cropSizeX : centerX + cropSizeX);
11    scanImageCrop = scanImageFull(centerZ - cropSizeZ : centerZ +
        cropSizeZ, centerX - cropSizeX : centerX + cropSizeX);
12
13    cropWidth = cropSizeX*pixelSize;
14    cropHeight = cropSizeZ*pixelSize;
15
16    X = -cropWidth:pixelSize:cropWidth;
17    Z = 0:-pixelSize:-2 * cropHeight;
18
19    [xDomain, zDomain] = meshgrid (X, Z);
20 end

1 function [ X, Z, ValueCancer, ValueSave, ValueCT ] = getCTValue(
    xDomain, zDomain, cancer, save, patientCT )
2
3     L0 = 1;
4     SSD = 100;
5
6     dx = 0.125;

```

```

7     dz = 0.125;
8
9     %Half of horizontal area length
10    A = (size(xDomain, 2) - 1) * dx / 2.;
11    %Full area depth
12    B = (size(xDomain, 2) - 1) * dz;
13
14    X = zeros(size(-A:dx:A, 2));
15    Z = zeros(size(0:dz:B, 2));
16
17    i = 0;
18    for step_x = -A:dx:A
19        i = i + 1;
20        j = 1;
21        for step_z = 0:dz:B
22            X(i, j) = step_x*step_z/(SSD - L0) + step_x;
23            Z(i, j) = L0 - step_z;
24            j = j + 1;
25        end
26    end
27
28    ValueCT = griddata(xDomain(1, :), zDomain(:, 1), patientCT, X,
29                       Z);
30    %
31    ValueSave = griddata(xDomain(1, :), zDomain(:, 1), save, X, Z)
32               ;
33    ValueSave(isnan(ValueSave)) = 0;
34    ValueSave(ValueSave <= 0.9) = 0;
35    ValueSave(ValueSave > 0.9) = 1;
36    %
37    ValueCancer = griddata(xDomain(1, :), zDomain(:, 1), cancer, X
38                           , Z);

```

```

37     ValueCancer(isnan(ValueCancer)) = 0;
38     ValueCancer(ValueCancer <= 0.9) = 0;
39     ValueCancer(ValueCancer > 0.9) = 2;
40
41 end

1 function [ z_effValue ] = z_effFunc( RatioValue , X, Z )
2     [m, n] = size(RatioValue);
3     z_effValue = zeros(m, n);
4
5     for i = 1:m
6         dist=((X(i,1)-X(i,2))^2+(Z(i,1)-Z(i,2))^2)^(0.5);
7         sum = 0;
8         for j = 1:n
9             if j>1
10                 sum = sum + RatioValue(i,j-1)*dist;
11             end
12             z_effValue(i,j) = sum + (dist*RatioValue(i,j))/2;
13         end
14     end
15 end

1 function [ sigmaValue ] = sigmaCalc( X, Z, z_effValue , RatioValue ,
    scaterPowerH20 , T_0, R_p )
2
3     [m, n] = size(RatioValue);
4     sigmaValue = zeros(m, n);
5
6     z_eff_temp(:, 2:n) = z_effValue(:, 1:n-1);
7     ratioValue_temp(:, 2:n) = RatioValue(:, 1:n-1);
8
9     energy = T_0 .* (1 - z_eff_temp ./ R_p);
10    energy(:, 1) = 0;
11    scatterPower = fnval(scaterPowerH20 , energy) .*
        ratioValue_temp;

```

```

12
13     for i = 1:m
14         dist = sqrt((X(i, 1)-X(i, 2))^2 + (Z(i, 1)-Z(i, 2))^2);
15         M0 = 0; M1 = 0; M2 = 0;
16         for j = 1:n
17             sigma = 0;
18             if j>1
19                 M2 = M2 - 2*M1 + M0 + dist*scatterPower(i, j);
20                 M1 = M1 - M0 - dist*scatterPower(i, j);
21                 M0 = M0 + dist*scatterPower(i, j);
22                 sigma = ( M2 + M1 + (M0/3))*dist*dist;
23             end
24             sigmaValue(i, j) = sigma;
25         end
26     end
27 end

1 function [ probValue, sigma_medValue, integralValue ] =
    prob_sigmaMed_integralValues( sigmaValue, X, Z, L0,
    tangentShadow, center, intSteps )
2
3     [m, n] = size(sigmaValue);
4     b = (size(X, 2) - 1) * 0.125 / 2.;
5     a = -b;
6
7     integralValue = zeros(m, n);
8
9     sigma_medValue = (abs(Z - L0) .* (tangentShadow * 0.391 *
    0.25)).^2 + sigmaValue;
10
11     probValue = exp(-(center - X).^2 ./ (2 .* sigma_medValue)) ./
    (2 * pi .* sigma_medValue).^0.5;
12     probValue(isnan(probValue)) = 0;
13     probValue(probValue > 1) = 1;

```

```

14
15     for i=1:m
16         for j=1:n
17             sum = 0;
18             for k = 1:intSteps-1
19                 val = a + (k*(b-a)/intSteps);
20                 sum = sum + gaussian(sigma_medValue(i, j), val, X(
                    i, j));
21             end
22
23             integralValue(i, j) = 1 / (2*pi*sigma_medValue(i, j))
                ^0.5*...
24             ((sum+(gaussian(sigma_medValue(i, j), a, X(i, j))
                +...
25             gaussian(sigma_medValue(i, j), b, X(i, j))))/2)*...
26             (b-a)/intSteps);
27             if isnan(integralValue(i, j))
28                 integralValue(i, j) = 0;
29             end
30             if integralValue(i, j) > 1
31                 integralValue(i, j) = 1;
32             end
33         end
34     end
35 end

1 function [ ValueDoseMeas, ValueZStep ] = doseCalc( X, Z, L_0, SSD,
    tangentShadow, doseMeasured, sigmaValue, intSteps )
2
3     [m, n] = size(X);
4     dz = 0.125;
5     b = (size(X, 2) - 1) * 0.125 / 2.;
6     a = -b;
7     midLengthPosition = m/2 + 0.5;

```

```

8     zeroDepthPosition = L_0/dz;
9
10    ValueZStep= zeros(1, n-zeroDepthPosition);
11    z_step = abs(Z(midLengthPosition, 1:n-zeroDepthPosition) - L_0
12               );
13
14    ValueDoseMeas = ((SSD + z_step) ./ SSD).^2 .* fnval(
15                   doseMeasured, z_step) ./ 100 - 0.05;
16
17
18    for j = 1:n-zeroDepthPosition
19        z = abs(Z(midLengthPosition, j) - L_0);
20        x = X(midLengthPosition, zeroDepthPosition+j);
21
22        sigmaM = (abs(Z(midLengthPosition, zeroDepthPosition+j) -
23                     L_0) *...
24                  tangentShadow * 0.391 * 0.25)^2 + sigmaValue(
25                      midLengthPosition, zeroDepthPosition+j);
26
27        sum = 0;
28        for k = 1:intSteps-1
29            value = a + (k * (b-a) / intSteps);
30            sum = sum + gaussian(sigmaM, value, x);
31        end
32
33        answer = 1/((2*pi*sigmaM)^0.5) *...
34                ((sum + (gaussian(sigmaM, a, x) + gaussian(sigmaM, b, x
35                    ))/2)*(b-a)/intSteps);
36
37        if isnan(answer)
38            answer = 1;
39        end
40
41        ValueZStep(1, j) = z;
42        ValueDoseMeas(1, j) = ValueDoseMeas(1, j) / answer;

```

```

36
37         if z > 3.5 || ValueDoseMeas(1, j) < 0
38             ValueDoseMeas(1, j) = 0;
39         end
40
41     end
42 end

1 function [ doseValue ] = doseValueCalc(z_effValue, integralValue
    , ...
2     doseMeasSpline, SSD )
3
4     doseValue = integralValue .* fnval(doseMeasSpline, z_effValue)
        .* (SSD ./ (SSD + z_effValue)).^2;
5     doseValue(doseValue < 0.1) = 0;
6     doseValue = doseValue .* 100;
7 end

1 function [ criticalScore, cancerScore, score ] = getScores(
    saugaValue, vezysValue, doseValue, cancerGoal, critGoal,
    otherGoal )
2
3     cancerScore=0;
4     criticalScore=0;
5     cancerNumber=0;
6     critNumber=0;
7     otherNumber = 0;
8     otherScore = 0;
9
10    [m, n] = size(doseValue);
11    for i = 1:m
12        for j = 1:n
13            flag = 0;
14            if saugaValue(i, j) > 0
15                critNumber = critNumber+1;

```

```

16         %criticalScore = criticalScore + ((doseValue(i,j)
           - critGoal)^2)^0.5;
17         criticalScore = criticalScore + doseValue(i,j);
18         flag = 1;
19     end
20     if vezysValue(i,j) > 1
21         cancerNumber = cancerNumber + 1;
22         %cancerScore = cancerScore + ((doseValue(i,j) -
           cancerGoal)^2)^0.5;
23         cancerScore = cancerScore + doseValue(i,j);
24         flag = 1;
25     end
26     if flag == 0
27         otherNumber = otherNumber + 1;
28         %otherScore = otherScore + ((doseValue(i,j) -
           otherGoal)^2)^0.5;
29         otherScore = otherScore + doseValue(i,j);
30     end
31 end
32 end
33
34 criticalScore = 1*criticalScore/critNumber;
35 cancerScore = -1*cancerScore/cancerNumber;
36 otherScore = 1*otherScore/otherNumber;
37
38 score = criticalScore + cancerScore + otherScore;
39 end

1 function [ edge ] = getSkinEdge( CTImage )
2
3     imSize = size(CTImage);
4     edgeMatrix = zeros(imSize);
5     for i = 1:imSize(1)
6         flag = 0;

```

```

7         j = 1;
8         while (j < imSize(2)) && (flag < 1)
9             %2 consecutive points required in case CT of rounding
               errors
10            if CTImage(i , j) > -248 && CTImage(i , j+1) > -248
11                edgeMatrix(i , j) = 1;
12                edgeMatrix(i , j + 1) = 1;
13                edgeMatrix(i , j + 2) = 1;
14                flag = 1;
15            end
16            j=j+1;
17        end
18    end
19    edge=edgeMatrix;
20 end

1 function [ BolusPoints , edgePoints ] = getBolusPoints( X, Z, edge ,
    numberOfPoints)
2     [m, n] = size(X);
3     func1 = csapi(X(:,1) , 1:m);
4     %%
5     A = 3;
6     bolusPoint = [0 , 0];
7     bolusPoints = zeros(numberOfPoints , 2);
8
9     distanceBetweenPoints= 2*A/numberOfPoints;
10
11    for i = 0:numberOfPoints
12        bolusPoint(1) = -A+distanceBetweenPoints*i;
13        eile = fnval(func1 , bolusPoint(1));
14        bolusPoint(2) = Z(eile , fixEdge(edge(eile , :)));
15        bolusPoints(i+1,:) = bolusPoint;
16    end
17

```

```

18     BolusPoints = bolusPoints;
19     edgePoints = checkBolus(X, Z, edge);
20 end

1 function [m, b] = getBolusLines(numberOfPoints, Points)
2     bolusLineM = zeros(1, numberOfPoints);
3     bolusLineB = zeros(1, numberOfPoints);
4
5     for i=1:numberOfPoints
6         bolusLineM(i) = (Points(i+1, 2)-Points(i, 2))/(Points(i+1,
            1)-Points(i, 1));
7         bolusLineB(i) = Points(i, 2)-bolusLineM(i)*Points(i, 1);
8     end
9     m = bolusLineM;
10    b = bolusLineB;
11 end

1 function y = checkPointPosition(x, z, edgePoints,dx)
2     n = size(edgePoints, 1);
3     y = 0;
4     for i = 1:n
5         if(dx >= abs(x-edgePoints(i,1))) && z > edgePoints(i,2)
6             y = 1;
7             break;
8         end
9     end
10 end

1 function [ solution ] = SimulatedAnealing( X, Z, RatioValue,
    saugaValue, vezysValue, ...
2     bolusPoints, numberOfPoints, edgePoints, ...
3     scaterPowerH20, tangentShadow, deapthDose, score, ...
4     dx, dz, center, intSteps, ...
5     L0, T_0, R_p, SSD, ...
6     cancerGoal, critGoal, otherGoal)

```

```

7
8     [m, n] = size(RatioValue);
9     H2OScatterRatio = 1;
10
11     maxSteps = 5;
12     currentStep = 1;
13     T0 = 1;
14     T = T0;
15     Tf = 1e-3;
16     alfa = 0.95;
17
18     guessBolusPoints = bolusPoints;
19
20     bestScore = score;
21     bestBolusPoints = bolusPoints;
22     %bestRatioValue = RatioValue;
23
24     tic
25     while (T > Tf) && (currentStep <= maxSteps)
26         fprintf('Iteration no: %i \n', currentStep);
27         heightToAdd = round(rand*1) +1;
28         basePoint = 6 + round(rand*7);
29         check = guessBolusPoints(basePoint, 2) + heightToAdd*dz;
30
31         if (check >= L0)
32             guessBolusPoints(basePoint, 2) = L0;
33         else
34             guessBolusPoints(basePoint, 2) = bestBolusPoints(
35                 basePoint, 2) + heightToAdd*dz;
36         end
37
38     [bolusLineM, bolusLineB] = getBolusLines(numberOfPoints,
39         guessBolusPoints);

```

```

38     guessRatioValue = RatioValue;
39
40     for i = 1:m
41         for j = 1:n
42             if checkPointPosition(X(i,j), Z(i,j), edgePoints,
43                                     dx) > 0
44                 flag = 0;
45                 k = 1;
46                 while k < numberOfPoints && flag == 0
47                     if guessBolusPoints(k, 1) <= X(i,j) &&
48                         guessBolusPoints(k+1,1) >= X(i,j)
49                         if ( bolusLineM(k) * X(i,j) +
50                             bolusLineB(k) - Z(i,j)) >=0
51                             guessRatioValue(i,j) =
52                                 H2OScatterRatio;
53                             flag = 1;
54                         end
55                     end
56                     k = k+1;
57                 end
58             end
59         end
60     end
61
62     z_effValue = z_effFunc(guessRatioValue, X, Z );
63     sigmaValue = sigmaCalc( X, Z, z_effValue, guessRatioValue,
64                             scaterPowerH20, T_0, R_p );
65     [probValue, sigma_medValue, integralValue] = ...
66         prob_sigmaMed_integralValues( sigmaValue, X, Z, L0,
67                                         tangentShadow, center, intSteps );
68
69     [ValueDoseMeas, ValueZDose] = doseCalc(X, Z, L0, SSD,
70                                             tangentShadow, deapthDose, sigmaValue, intSteps);

```

```

64     doseMeasSpline = csapi( ValueZDose , ValueDoseMeas );
65     doseValue = doseValueCalc( z_effValue , integralValue ,
        doseMeasSpline , SSD );
66
67     [critOld , cancerOld , guessScore] = getScores( saugaValue ,
        vezysValue , doseValue , cancerGoal , critGoal , otherGoal)
        ;
68
69     df = guessScore - bestScore;
70     if df < 0
71         bestScore = guessScore;
72         bestBolusPoints = guessBolusPoints;
73         %bestRatioValue = guessRatioValue;
74         bestDose = doseValue;
75     elseif (rand < probability(df, T) && df > 1e-2)
76         bestScore = guessScore;
77         bestBolusPoints = guessBolusPoints;
78     end
79     fprintf( 'best score: %f \n' , bestScore );
80     T=T0*alfa^currentStep;
81     currentStep = currentStep + 1;
82     fprintf( '-----\n' );
83
84     end
85     toc
86
87     solution = bestDose;
88 end

1 function [ candidate ] = GeneticOptimization( N, X, Z, ratioValue ,
    deapthDose , ...
2     H2OScatterRatio , scatterPowerH20 , ...
3     skinEdge , edgePoints , A, numPoints , dx, dz , L0, SSD, T_0, R_p,
    ...

```

```

4     tangentShadow , center , intSteps , ...
5     criticalValue , tumorValue , tumorGoal , criticalGoal , otherGoal)
6     tic
7
8     funci = csapi(X(:,1) , 1: size(X, 1));
9     edgeInterpol = csapi(edgePoints(:,1) , edgePoints(:,2));
10
11     distanceBetweenPoints= 2*A/numPoints;
12
13     bolusArray(1:N) = GA_Bolus;
14     % generating initial bolus population
15     for i = 1:N
16         bolusPoint = [0 , 0];
17         bolusPoints = zeros(numPoints , 2);
18         for j = 0:numPoints
19             bolusPoint(1) = -A + (distanceBetweenPoints*j);
20             eile = fnval(funci , bolusPoint(1));
21             bolusPoint(2) = Z(eile , fixEdge(skinEdge(eile , :)));
22
23             bolusPoint(2) = bolusPoint(2) + dz*(round((rand*(L0 -
24                 bolusPoint(2)))/dz)-1);
25             bolusPoints(j+1,:) = bolusPoint;
26         end
27         bolusArray(i).BolusPoints = bolusPoints;
28
29         bolusRatioValue = getBolusRatio(X, Z, H2OScatterRatio ,
30             ratioValue , bolusArray(i).BolusPoints , ...
31             edgePoints , dx );
32
33         doseDistribution = getDoseDistribution(X, Z,
34             bolusRatioValue , scatterPowerH20 , ...
35             deapthDose , L0 , SSD , T_0 , R_p , tangentShadow , center ,
36             intSteps);

```

```

33
34
35     [critOld, cancerOld, bolusScore] = getScores(
        criticalValue, tumorValue, doseDistribution, ...
36         tumorGoal, criticalGoal, otherGoal);
37
38     bolusArray(i).Fitness = bolusScore;
39 end
40
41 for i = 1:N
42     %figure
43     %scatter(bolusArray(i).BolusPoints(:,1), bolusArray(i).
        BolusPoints(:,2))
44     fprintf('Fitness: %f\n', bolusArray(i).Fitness)
45 end
46
47     currentPopulation = bolusArray;
48     newPopulation(1:N) = GA_Bolus;
49     fitnessSort = zeros(1, N);
50 for i = 1:5
51     fprintf('Generation: %i\n', i)
52     if N > 3
53         %selecting
54         for j = 1:N
55             fitnessSort(j) = currentPopulation(j).Fitness;
56         end
57         fitnessSort = sort(fitnessSort);
58
59         for j = 1:round(N/2)
60             for k = 1:N
61                 if fitnessSort(j) == currentPopulation(k).
                    Fitness
62                     newPopulation(j) = currentPopulation(k);

```

```

63             break ;
64         end
65     end
66 end
67 %crossing
68 for j = 1:(round(N/2) - 1)
69     newPopulation(round(N/2)+j).BolusPoints = vertcat
70         (...
            newPopulation(j).BolusPoints(1:round(size(
                newPopulation(j).BolusPoints, 1)/2),:),
            ....
            newPopulation(j+1).BolusPoints(round(size(
                newPopulation(j+1).BolusPoints, 1)/2)+1:
                size(newPopulation(j+1).BolusPoints, 1),:))
            ;
71
72 %Mutate
73 for k = 1:numPoints
74     if rand < 0.1
75         newPopulation(round(N/2)+j).BolusPoints(k,
76             2) = ...
77             fnval(edgeInterpol, newPopulation(
78                 round(N/2)+j).BolusPoints(k, 1)) +
79                 ...
80                 rand*(L0 - fnval(edgeInterpol,
81                     newPopulation(round(N/2)+j).
82                     BolusPoints(k, 1)));
83     end
84 end
85
86 %Calculate fitness
87 bolusRatioValue = getBolusRatio(X, Z,
88     H2OScatterRatio, ratioValue, ...
89     newPopulation(round(N/2)+j).BolusPoints,

```

```

        edgePoints , dx);
84     doseDistribution = getDoseDistribution(X, Z,
        bolusRatioValue , scatterPowerH20 , ...
85     deapthDose , L0, SSD, T_0, R_p, tangentShadow ,
        center , intSteps);
86     [critOld , cancerOld , bolusScore] = getScores(
        criticalValue , tumorValue , doseDistribution ,
        ...
87     tumorGoal , criticalGoal , otherGoal);
88
89     newPopulation(round(N/2)+j).Fitness = bolusScore;
90     newPopulation(round(N/2)+j).DoseValue =
        doseDistribution;
91     newPopulation(round(N/2)+j).RatioValue =
        bolusRatioValue;
92     end
93
94     if rem(N, 2) == 0
95         newPopulation(N).BolusPoints = vertcat(...
96             currentPopulation(N).BolusPoints(1:round(size(
                currentPopulation(N).BolusPoints , 1)/2) ,:)) ,
                ....
97             currentPopulation(1).BolusPoints(round(size(
                currentPopulation(1).BolusPoints , 1)/2)+1:
                size(currentPopulation(1).BolusPoints , 1)
                ,:));
98
99     bolusRatioValue = getBolusRatio(X, Z,
        H2OScatterRatio , ratioValue , ...
100     newPopulation(N).BolusPoints , edgePoints , dx);
101     doseDistribution = getDoseDistribution(X, Z,
        bolusRatioValue , scatterPowerH20 , ...
102     deapthDose , L0, SSD, T_0, R_p, tangentShadow ,

```

```

        center , intSteps);
103         [critOld , cancerOld , bolusScore] = getScores(
            criticalValue , tumorValue , doseDistribution ,
            ...
104         tumorGoal , criticalGoal , otherGoal);
105         newPopulation(N).Fitness = bolusScore;
106         newPopulation(N).DoseValue = doseDistribution;
107         newPopulation(N).RatioValue = bolusRatioValue;
108     end
109     currentPopulation = newPopulation;
110
111     for j = 1:N
112         %figure
113         %scatter(currentPopulation(j).BolusPoints(:,1) ,
            currentPopulation(j).BolusPoints(:,2))
114         fprintf('Fitness: %f\n' , currentPopulation(j).
            Fitness)
115     end
116     else
117         fprintf('Need more samples')
118     end
119 end
120
121 for j = 1:N
122     fitnessSort(j) = currentPopulation(j).Fitness;
123 end
124 fitnessSort = sort(fitnessSort);
125 for k = 1:N
126     if fitnessSort(1) == currentPopulation(k).Fitness
127         bestCandidate = currentPopulation(k);
128         break;
129     end
130 end

```

```
131     toc
132     candidate = bestCandidate;
133 end

1  classdef GA_Bolus
2
3     properties
4         BolusPoints
5         RatioValue
6         Fitness
7         DoseValue
8     end
9
10    methods
11    end
12
13 end
```