DIDACTICA MATHEMATICA, Vol. 42 (2024) , pp. 01–32

OPTIMIZAREA LEXICOGRAFICĂ ÎN PLANIFICAREA TRATAMENTULUI CU IMRT

Iulia-Antonia A. Bartic

Abstract. This thesis aims to provide an overview of lexicographic optimization, an efficient tool for handling the distribution of radiation doses in Intensity-Modulated Radiation Therapy (IMRT) planning for cancer patients. Beginning with the definition of elementary notions, we will study what constitutes a lexicographic optimal solution for a minimization vectorial problem, its existence, and methods for finding it using the Cutting Plane method and scalarization. Subsequently, we will translate IMRT planning for prostate cancer into the specific language of lexicographic optimization and, following the results and optimization system used in practice, implement our own system. This system is a program that utilizes principles from the Cutting Plane method for solving lexicographic optimization problems and the scalarization approach to find an optimal plan for radiation dose distribution.

Key words. lexicographic optimization, IMRT planning, Cutting plane method, scalarization.

1. INTRODUCERE

1.1. Ordonarea lexicografică în $\mathbb{R}^n$.

În spaţiul vectorial $\mathbb{R}^n$, deoarece nu putem compara orice doi vectori, avem o ordine parţială, iar o modalitate de introducere a relaţiilor de ordine parţiale este folosind conuri convexe.

1.1.1. *Conul lexicografic.*

Definiţia 1. *Numim* ***con lexicografic*** *şi notăm $\mathcal{C}_{lex}$ o submulţime a lui $\mathbb{R}^n$ care conţine toţi vectorii pentru care primul coeficient nenul este strict pozitiv:*

$$\mathcal{C}_{lex} = \{0_n\} \cup \left\{x \in \mathbb{R}^n \mid x_i = 0,\ i = \overline{1,k},\ x_{k+1} > 0,\ \forall\, k \in \mathbb{N},\ 1 \leq k < n\right\},$$

unde $x = (x_1, \ldots, x_n)$.

Propoziţia 1. *Conul lexicografic are mai multe proprietăţi:*

- *$\mathcal{C}_{lex}$ nu este un con propriu;*
- *$\mathcal{C}_{lex}$ este o mulţime convexă;*
- *$\mathcal{C}_{lex}$ este ascuţit deoarece:*

$$l(\mathcal{C}_{lex}) = \mathcal{C}_{lex} \cap (-\mathcal{C}_{lex}) = \{0\};$$

Coordonator ştiinţific: Lect. univ. dr. Anca Grad

- $\mathcal{C}_{lex}$ *nu este corect deoarece:*

$$\mathrm{cl}(\mathcal{C}_{lex}) + \mathcal{C}_{lex} \setminus l(\mathcal{C}_{lex}) \not\subset \mathcal{C}_{lex};$$

- *mulţimea* $\mathcal{C}_{lex}$ *nu este nici închisă, nici deschisă.*

1.1.2. *Ordonarea lexicografică indusă de conul lexicografic .*

Conul lexicografic induce pe $\mathbb{R}^n$ o ordine liniară. Astfel, dacă $x, y \in \mathcal{C}_{lex}$ sortarea indusă de con se realizează după cum urmează:

$$x \preceq_{lex} y \Leftrightarrow y - x \in \mathcal{C}_{lex}.$$

Definiţia 2. *Ordinea lexicografică, notată* $\preceq_{lex}$*, este o relaţie de ordine definită pe* $\mathbb{R}^n$*, care compară pe rând elementele secvenţelor (vectorilor de comparat), de la început la sfârşit.*

Fie secvenţele $s_1 = (x_1, \ldots, x_n) \in \mathbb{R}^n$ şi $s_2 = (y_1, \ldots, y_n) \in \mathbb{R}^n$. Avem că $s_1 \preceq_{lex} s_2$ dacă este îndeplinită una din condiţiile următoare:

(1) Dacă $n = 0 \Rightarrow$ s_1 şi s_2 sunt considerate egale.
(2) Dacă $n > 0$ şi $\exists\, k \in \mathbb{N}$, cu $1 \leq k \leq n$ a.î. $x_i = y_i,\ \forall 1 \leq i < k$, iar $(x_k, \ldots, x_n) \preceq_{lex} (y_k, \ldots, y_n)$.

Exemplul 1. *Comparaţi următoarele secvenţe:* $(1, 2)$ *şi* $(1, 2)$*;* $(1, 2, 3)$ *şi* $(1, 3, 2)$*;* $(1, 2, 2, 5)$ *şi* $(1, 2, 1, 5)$.

Soluţia 1.

$$(1, 2) \preceq_{lex} (1, 2)$$
$$(1, 2, 3) \preceq_{lex} (1, 3, 2)$$
$$(1, 2, 2, 5) \not\preceq_{lex} (1, 2, 1, 5).$$

În continuare, vom folosi tot notaţia "$\leq$" pentru compararea vectorilor din $\mathbb{R}^n$, în loc de "$\preceq_{lex}$".

Pentru o abordare mai aprofundată a noţiunilor de topologia spaţiului $\mathbb{R}^n$, noţiuni legate de funcţii vectoriale şi elemente de bază din Teoria Optimizării, vezi [13], [6], [14] şi [10].

2. OPTIMIZAREA LEXICOGRAFICĂ

Problemele de optimizare lexicografică apar atunci când avem de rezolvat mai multe obiective conflictuale, ce pot fi ierarhizate în funcţie de importanţa lor.

Materialul prezentat în acestă secţiune a fost documentat în principal din articolul [9], dar şi din următoarele referinţe bibliografice: [5], [11], [15], [16] şi [17].

2.1. Formularea problemei .

Fie L multimea tuturor soluţiilor optime lexicografic a unei probleme date. Vom defini o relaţie binară pentru ordonarea lexicografică a doi vectori z = $(z_1, z_2, ..., z_\ell)$ şi z' = $(z'_1, z'_2, ..., z'_\ell)$ din $\mathbb{R}^\ell$ astfel:

$$z \leq^L z' \Leftrightarrow ((z = z') \text{ sau } (\exists j \in N_\ell \text{ a.î. } \forall i \in N_{j-1} : z_i = z'_i,\ z_j < z'_j)),$$
$$\text{unde } N_0 = \emptyset, \text{iar } N_\ell = \{1, 2, \ldots, \ell\}.$$

Considerăm următoarea problemă de optimizare:

$$Z_L(F, X) : \min^L\{F(x) \mid x \in X\}, \text{unde} \tag{1}$$

$$F(x) = (f_1(x), f_2(x), \ldots, f_\ell(x)), \quad \ell \geq 2,$$
$$f_k(x) = \langle c_k, x\rangle, \quad c_k \in \mathbb{R}^n, \quad k \in N_\ell = \{1, 2, \ldots, \ell\},$$
$$X = \left\{x \in \mathbb{R}^n \mid g^i(x) \leq 0,\ x \geq 0,\ i \in N_m\right\}, \quad X \neq \emptyset,$$
$$\text{iar } g^i(x) \text{ sunt funcţii convexe } \forall i \in N_m, \text{ cu } n, m \in \mathbb{N}.$$

Definiţia 3. *Vectorul x este* ***preferabil lexicografic*** *vectorului x' dacă este îndeplinită una dintre următoarele ℓ condiţii:*

1) $f_1(x) < f_1(x')$;
2) $f_1(x) = f_1(x'),\ f_2(x) < f_2(x')$;
$\vdots$
l) $f_j(x) = f_j(x'),\ j \in \{1, 2, ..., l-1\},\ f_\ell(x) < f_\ell(x')$.

Definiţia 4. *Doi vectori x şi x' sunt* ***echivalenţi*** *dacă pentru fiecare criteriu, vectorii au aceleaşi estimări ($x \neq x'$).*

$$\Leftrightarrow f_i(x) = f_i(x'), \forall i \in \{1, 2, \ldots, \ell\}.$$

Definiţia 5. ***Mulţimea soluţiilor optime lexicografic*** *ale problemei $Z_L(F, X)$ se defineşte în felul următor:*

$$L(F, X) = \{x \in X \mid \nu(x, F, X) = \emptyset\}, \text{ unde}$$

$$\nu(x, F, X) = \{x' \in X \mid \exists j \in N_\ell : f_j(x) < f_j(x') \text{ sau}$$
$$j = \min\{i \in N_\ell : f_i(x) \neq f_i(x')\}\}.$$

Conform definiţiei precedente, putem observa că acestea se pot determina şi direct, utilizând relaţiile recurente

$$L_i(F, X) = \text{argmin}\{x \in X \mid f_i(x) = \min\{f_i(x) | x \in L_{i-1}(F, X)\}\}, i \in N_\ell \tag{2}$$

unde argmin$\{\cdot\}$ este mulţimea soluţiilor optime corespunzătoare problemei de minimizare, iar

$$L_0(F, X) = X,$$
$$L_\ell(F, X) = L(F, X).$$

Considerând recurenţa (2), deducem următoarul şir de incluziuni:

$$X = L_0(F, X) \supseteq L_1(F, X) \supseteq L_2(F, X) \supseteq \ ... \ \supseteq L_\ell(F, X) = L(F, X),$$

o consecinţă imediată fiind faptul că mulţimea L(F, X) poate fi determinată rezolvând secvenţa de ℓ probleme scalare de programare convexă Z_{L_i}, $i \in N_\ell$.

Propoziţia 2. *În cazul problemelor Z_{L_i}, $i \in N_\ell$, orice punct de minim (maxim) local este şi punct de minim (maxim) global.*

Pentru a defini soluţiile optime lexicografic trebuie să avem în vedere respectarea unor proprietăţi:

(1) Pentru $\forall x^0 \in X$, care îndeplineşte $f_1(x^0) < f_1(x)$ atunci când $x \in X \setminus \{x^0\}$ este o soluţie, avem că $x^0 \in L(F, X)$.
(2) Dacă $x \in X$ este o soluţie şi $\exists\, x' \in X \setminus \{x\}$ astfel încât $f_1(x) > f_1(x')$, atunci $x \notin L(F, X)$.

Definiţia 6. *Numim **soluţie optimă lexicografic** o soluţie $x^* \in X$ a problemei $Z_L(F, X)$ care nu este mai rea (nu este mai mare lexicografic) decât alte soluţii admise $y \in X \Leftrightarrow$*

$$F(x^*) \text{ - } F(y) \leq^L 0 \Leftrightarrow F(y) \text{ - } F(x^*) \geq^L 0.$$

Astfel, putem spune că $x \in X$ este o soluţie optimă lexicografic dacă:

$$x \in L(F, X) \Leftrightarrow \{y \in X \mid F(y) <^L F(x)\} = \emptyset.$$

În contextul problemelor de optimizare lexicografică, alegerea celei mai bune soluţii se face în funcţie de următoarea regulă: se va alege o soluţie care va aduce o îmbunătăţire la criteriile mai importante, în detrimentul oricăror pierderi în criteriile mai puţin importante.

2.2. Existenţa soluţiilor optime lexicografic .

Existenţa soluţiilor optime a problemei de optimizare lexicografică din mulţimea soluţiilor posibile X e condiţionată de proprietăţile relaţiei de ordine folosite, de structura lui X şi a elementelor sale, etc. În continuare vom face câteva observaţii şi vom da câteva exemple pentru a ne familiariza cu problema.

Observaţia 1. *Cu privire la existenţa acestor soluţii putem observa câteva aspecte:*

- *O condiţie suficientă pentru existenţa soluţiilor optime lexicografic este ca mulţimea X să fie finită.*

Exemplul 2. *Fie $F : X \to \mathbb{R}^3$, $F(x) = (f_1(x), f_2(x), f_3(x))$, unde f_1, f_2, f_3 sunt definite în felul următor:*

$$f_1, f_2, f_3 : \mathbb{R} \to \mathbb{R},\ f_1(x) = x,\ f_2(x) = x - 2,\ f_3(x) = x + 1,$$

iar $X = \{1, -1, -3, 0\}$. Găsiţi soluţia optimă lexicografic pentru problema de minimizare.

Soluţia 2. *Calculând valorile lui F în fiecare punct din X şi ordonând lexicografic rezultatele obţinute, avem:*

$$\textbf{\textit{(-3, -5, -2)}} \leq (-1,-3,0) \leq (0,-2,1) \leq (1,-1,2) \Leftrightarrow$$
$$\Leftrightarrow \quad F(x_3) \leq F(x_2) \leq F(x_4) \leq F(x_1)$$
$$\Rightarrow x_3 = -3 \text{ este soluţia optimă lexicografic.}$$

- *Existenţa soluţiilor optime lexicografic este asigurată dacă mulţimea $S = \{F(x)\text{: } x \in X\}$ este o mulţime mărginită şi închisă.*

Exemplul 3. *Fie $F : X \to \mathbb{R}^2$, $F(x,y) = (f_1(x,y), f_2(x,y))$, unde*

$$f_1, f_2 : \mathbb{R}^2 \to \mathbb{R},\ f_1(x,y) = x^2,\ f_2(x,y) = x,$$

iar $X = [0,3] \times [0,2]$ - mulţime mărginită şi închisă. Găsiţi soluţia optimă lexicografic pentru problema de minimizare.

Soluţia 3. *Funcţia f_1 îşi atinge minimul în toate punctele de forma*

$$S_1 = \{(0,y) \mid y \in [0,2]\}, \quad f_1(0,y) = 0^2 = 0.$$

Am restrâns spaţiul de căutare al soluţiilor de la S la S_1 şi observăm că şi f_2 îşi atinge minimul în toate punctele din S_1, adică

$$f_2(x_1,x_2) = f_2(0,x_2) = 0,\ \forall\,(x_1,x_2) \in S_1.$$

În acest caz, mulţimea soluţiilor posibile este

$$S_1 = \{(0,y) : y \in [0,2]\},$$

iar soluţia optimă lexicografic este $(0,0)$
deoarece, ordonând soluţiile lexicografic obţinem:
$$(0,0) \leq \cdots \leq (0,1) \leq \cdots \leq (0,2).$$

- *Dacă mulţimea în care căutăm soluţii este infinită şi nu avem alte restricţii pentru ea, nu putem garanta existenţa soluţiilor optime din punct de vedere lexicografic.*

Exemplul 4. *Fie $F : X \to \mathbb{R}^2$, $F(x,y,z) = (f_1(x,y,z), f_2(x,y,z))$, unde*

$$f_1, f_2 : \mathbb{R}^3 \to \mathbb{R},\ f_1(x,y,z) = x,\ f_2(x,y,z) = y + z,$$

iar $X = \mathbb{R} \times \{1\} \times \{3\}$. Studiaţi existenţa soluţiei optime lexicografic pentru problema de minimizare.

Soluţia 4. *Deoarece $y \in \{1\}$ şi $z \in \{3\}$, adică $y = 1$ şi $z = 3$, avem că*

$$f_2(x,y,z) = y + z = 1 + 3 = 4,\ \forall (x,y,z) \in X.$$

Imaginea funcţiei $f_1(x,y,z) = x$ este întreg $\mathbb{R}$ când $(x,y,z) \in X$, de unde deducem că nu vom putea determina o valoare minimă a acesteia, deci nici pentru funcţia F nu vom putea determina o valoare minimă. În acest caz, nu avem soluţii optime lexicografic pentru problema minimizării lui F.

În contextul observaţiilor precedente, deducem că ar fi relevantă studierea existenţei soluţiilor pentru problema de optimizare lexicografică pe mulţimi nemărginite, adăugând unele constrângeri, cum ar fi ca mulţimea să fie şi convexă.

Fie $0^+X = \{d \in \mathbb{R}^n \mid \forall\, x \in X : \quad x + \alpha d \in X, \alpha \geq 0\}$ conul de recesiune al mulţimii X.

Propoziţia 3. *Dacă X este o mulţime convexă şi nemărginită, atunci conul său de recesiune va fi o mulţime nevidă.*

$$\Leftrightarrow 0^+X \setminus \{0\} \neq \emptyset.$$

Conul de recesiune $K^L = \left\{x \in \mathbb{R}^n \mid Cx <^L 0\right\}$ este un con convex de direcţii ale vectorilor lexicografic negativi, care determină ordinea lexicografică în spaţiul $\mathbb{R}^\ell$. Acesta poate fi reprezentat ca o reuniune de mulţimi disjuncte:

$$K^L = K_1 \cup K_2 \cup \cdots \cup K_\ell, \tag{3}$$

unde mulţimile $K_1,\ K_2,\ \ldots,\ K_\ell$ sunt definite astfel:

$K_1 = \{x \in \mathbb{R}^n \mid c_1 x < 0\}$,

$K_2 = \{x \in \mathbb{R}^n \mid c_1 x = 0,\ \ c_2 x < 0\}$,

$\ldots$,

$K_\ell = \{x \in \mathbb{R}^n \mid c_1 x = 0,\ \ c_2 x = 0,\ \ \ldots,\ \ c_{\ell-1} x = 0,\ \ c_\ell x < 0\}$.

Vom studia problema $Z_L(F, X)$, folosindu-ne de proprietăţile conului de recesiune şi de ordonarea lexicografică dată de conul K^L, pe care îl vom numi şi conul de perspectivă al direcţiilor lexicografice ale problemei $Z_L(F, X)$.

Propoziţia 4. $x \in L(F, X) \Leftrightarrow (x + K^L) \cap X = \emptyset$

Demonstraţie. "$\Rightarrow$"

Fie x o soluţie optimă lexicografic a problemei de minimizare, adică $x \in L(F, X)$.

Presupunem, prin reducere la absurd, că $(x + K^L) \cap X \neq \emptyset \Rightarrow$

$$\Rightarrow \exists\, y \in (x + K^L) \cap X$$

$$\Rightarrow y = x + \alpha d, \text{ unde } d \in K^L, \alpha \geq 0, \quad \text{şi} \quad y \in X.$$

Din definiţia conului K^L ştim că d este o direcţie în care dacă ne deplasăm, obţinem o scădere a valorilor funcţiilor obiectiv în sens lexicografic. Aceasta intră în contradicţie cu ideea de la care am pornit, că x este o soluţie optimă lexicografic ($\Leftrightarrow$ nu există soluţii mai bune decât x care să minimizeze F)$\Rightarrow$

$$(x + K^L) \cap X = \emptyset.$$

"$\Leftarrow$"

Presupunem că $(x + K^L) \cap X = \emptyset \Rightarrow$

$\Rightarrow$ nu avem puncte în X în care să putem ajunge printr-o direcţie din K^L, adică x nu poate fi îmbunătăţit lexicografic în nicio direcţie din K^L, încât să

rămână în X
⇒ x este o soluţie optimă lexicografic a problemei $Z_L(F, X)$, deci $x \in L(F, X)$.
□

Teorema 1. *O condiţie necesară pentru existenţa soluţiilor optime lexicografic pentru problema $Z_L(F, X)$ este ca intersecţia dintre conul K^L al direcţiilor lexicografice şi conul recesiv 0^+X să fie vidă:*

$$K^L \cap 0^+X = \emptyset. \tag{4}$$

Demonstraţie. Presupunem că mulţimea soluţiilor optime lexicografic $L(F, X) \neq \emptyset \Rightarrow K^L \cap 0^+X \neq \emptyset$, adică nu e îndeplinită condiţia (4).

Au loc următoarele relaţii:

$$(x + K^L) \cap X \supseteq (x + K^L) \cap (x + 0^+X) = x + (K^L \cap 0^+X) \neq \emptyset.$$

Conform propoziţiei (4), deducem că $L(F, X) = \emptyset$, contradicţie cu presupunerea iniţială.

□

Teorema 2. *Fie V o mulţime nevidă ce conţine punctele de extrem ale mulţimii convexe închise X. Dacă mulţimea V este mărginită, atunci mulţimea X are un minim lexicografic dacă şi numai dacă e mărginită în toate direcţiile lexicografic negative.*

Fie $P \subseteq \mathbb{R}^n$ o mulţime poliedrală, $A \in \mathbb{R}^{m \times n}$ o matrice $(n, m \in \mathbb{N})$ şi vectorul $b \in \mathbb{R}^n$.

$$P = \{x \in \mathbb{R}^n \mid Ax \leq b\}.$$

Propoziţia 5. *Mulţimile poliedrale sunt mulţimi convexe.*

Corolarul 1. *Mulţimea poliedrală şi convexă X are un minim lexicografic dacă şi numai dacă este mărginită în toate direcţiile lexicografic negative.*

Corolarul anterior şi teorema (1) implică următoarea teoremă:

Teorema 3. *Fie X o mulţime poliedrală, închisă şi convexă. Atunci condiţia (4), adică $K^L \cap 0^+X = \emptyset$, este o condiţie necesară şi suficientă pentru existenţa soluţiilor optime lexicografic pentru problema $Z_L(F, X)$.*

2.3. Condiţii de optimalitate a soluţiilor .

Vorbim despre conceptul de **optimalitate Pareto** atunci când în procesul de optimizare, avem obiective conflictuale (îmbunătăţirea unui obiectiv conduce inevitabil la degradarea altuia). O soluţie se numeşte Pareto-optimală dacă reprezintă cea mai bună variantă de compromis între obiectivele conflictuale.
Fie P(F, X) mulţimea tuturor soluţiilor Pareto-optimale ale problemei minimizării lui F şi fie $x \in X$.

$$x \in P(F, X) \Leftrightarrow \{y \in X \mid F(y) \leq F(x), F(y) \neq F(x)\} = \emptyset.$$

Pentru o abordare mai detaliată a soluţiilor Pareto-optimale, vezi [8] şi [12].

Dacă pentru o problemă de optimizare, obiectivele (criteriile) sunt la fel de importante, atunci putem vorbi despre **soluţii Slater-optimale**.

Notăm Sℓ(F, X) mulţimea tuturor soluţiilor Slater optimale ale problemei minimizării lui F şi fie $x \in X$.

$$x \in S\ell(F, X) \Leftrightarrow \{y \in X \mid F(y) < F(x)\} = \emptyset.$$

Pentru o abordare mai detaliată a soluţiilor Slater-optimale, vezi [7].

Observaţia 2. *Conform definiţiilor celor două tipuri de mulţimi de soluţii, au loc următoarele incluziuni:* $L(F, X) \subseteq P(F, X) \subseteq S\ell(F, X)$.

În cazul problemelor de optimizare vectorială, când criteriile sunt la fel de importante pentru luarea deciziilor, soluţiile optime pot fi privite ca o submulţime a uneia din cele două mulţimi $P(F, X)$ sau $S\ell(F, X)$.

Datorită liniarităţii funcţiilor criteriu şi a structurii mulţimii X, este binecunoscut în domeniul optimizării vectoriale că soluţiile Pareto-optimale şi cele Slater-optimale pot cuprinde întreaga mulţime de soluţii posibile sau sunt localizate doar pe frontiera acestei mulţimi. În continuare, având în vedere incluziunile $L(F, X) \subseteq P(F, X) \subseteq S\ell(F, X)$ în stabilirea condiţiilor necesare şi suficiente pentru optimalitatea lexicografică a soluţiilor problemei, vom considera doar mulţimea punctelor de pe frontiera lui X (bd(X)).

Vom defini următoarele mulţimi:

$$N(y) = \{i \in N_m \mid g_i(y) = 0\},$$
$$X(y) = \{x \in \mathbb{R}^n \mid g_i(x) \leq 0,\ i \in N(y)\}.$$

Dacă funcţiile $g_i(x)$, $i \in N(y)$ sunt diferenţiabile şi continue în $\mathbb{R}^n$, putem defini mulţimea

$$Q(y) = \{x \in \mathbb{R}^n \mid \langle \nabla g_i(y), x - y\rangle \leq 0,\ i \in N(y)\},$$

unde $\nabla g_i(y)$ este gradientul funcţiei g_i în punctul y, $i \in N(y)$. Reiese imediat că

$$y + 0^+X \ \subseteq X \ \subseteq X(y) \ \subseteq Q(y). \tag{5}$$

Teorema 4. *Fie* $y \in \text{bd}(X)$, $K^L = K_1 \cup K_2 \cup \ldots \cup K_\ell$,

$$K_1 = \{x \in \mathbb{R}^n \mid c_1x < 0\}, \ldots,$$
$$K_\ell = \{x \in \mathbb{R}^n \mid c_1x = 0,\ c_2x = 0,\ \ldots,\ c_{\ell-1}x = 0,\ c_\ell x < 0\}.$$

Dacă funcţiile g_i (funcţii convexe - constrângerile ce formează mulţimea soluţiilor admise X) sunt continue şi diferenţiabile, atunci relaţia

$$K^L \cap (Q(y) - y) = \emptyset \tag{6}$$

este o condiţie suficientă ca y să fie soluţie optimă lexicografic ($y \in L(C, X)$). Mai mult, dacă $\{\nabla g_i(y) : i \in N(y)\}$ este un sistem liniar de vectori independenţi, atunci relaţia

$$K_1 \cap (Q(y) - y) = \emptyset \tag{7}$$

este o condiţie necesară ca să aibă loc incluziunea $y \in L(C, X)$.

Demonstraţie. **Suficienţa.**
Din relaţia (5) avem că $y + 0^+X \subseteq Q(y)/-y$
$\Rightarrow 0^+X \subseteq Q(y) - y$ (*)
Din relaţia (4) a teoremei (1) ştim că $K^L \cap 0^+X = \emptyset$ (**)
Din (*) şi (**) rezultă imediat că formula (6) are loc.
Pentru demonstrarea necesităţii, vezi [9]. □

2.4. Metoda tăierii planului în rezolvarea problemelor de optimizare vectorială convexă folosind ordonarea lexicografică .

Căutarea soluţiilor problemei $Z_L(F, X) : \min^L\{F(x) \mid x \in X\}$ poate fi redusă la rezolvarea unui şir de probleme de optimizare liniară lexicografică.

$$Z_L(F, X_p) : \min^L\{F(x) \mid x \in X_p\},$$

unde $p \in \mathbb{N}$, iar X_p este o mulţime poliedrală dată de relaţia:

$$X_p = \left\{x \in \mathbb{R}^n \mid \langle \nabla g_i(x^j), : x - x^j\rangle + g_i(x^j) \leq 0,\ \ x \geq 0,\ \ i \in N_m,\ \ j = \overline{0, p}\right\},$$

$$x^j \in \mathbb{R}^n_+ = \{x \in \mathbb{R}^n \mid x_i \geq 0, i \in N_n\}.$$

Lema 1. *Are loc următoarea incluziune:* $X \subset X_p$.

Teorema 5. *Dacă funcţia vectorială* F *atinge o valoare de minim lexicografic, printre punctele în care această valoare de minim este atinsă, unele se află şi în* X_p.

Pornind de la această teoremă, deducem că pentru a enumera aceste puncte de extrem ale mulţimii X_p ce vor duce la rezolvarea problemei $Z_L(F, X_p)$, se poate aplica un algoritm simplex.

Găsirea soluţiilor optime lexicografic ale problemei $Z_L(F, X_p)$ se reduce astfel la rezolvarea problemei de minimizare $Z(f_s, X_p) : \min\{f_s(x) \mid x \in X_p\}$, $S \in N_\ell$, în care funcţia corespunzătoare vectorului criteriu ordonat lexicografic este minimiziată. Ideea de bază a acestui algoritm este următoarea: dacă o sluţie optimă a problemei $Z(f_s, X_p)$ nu este soluţie şi pentru problema $Z_L(F, X)$, aceasta va fi exclusă din mulţimea soluţiilor optime ale problemei $Z(f_s, X_p)$ prin adăugarea unei constrângeri. Această nouă constrângere taie o parte din X (o parte în care nu avem soluţii valide pentru problema $Z_L(f, X)$). Dacă soluţia optimă a problemei $Z(f_s, X_p)$ este din X şi este singura soluţie optimă pe această mulţime X, atunci soluţia găsită este soluţia optimă lexicografic a problemei $Z_L(f, X)$.

2.5. Algoritm pentru rezolvarea problemei $Z_L(F, X)$.

Fie $s = 1$, $k = 0$ şi fie $x^k \in \mathbb{R}^n$ ales arbitrar. Folosindu-ne de acestea, vom construi mulţimea poliedrală:

$$X_k = \left\{ x \in \mathbb{R}^n \mid \langle \nabla g_i(x^k), x - x^k \rangle + g_i(x^k) \leq 0, \;\; x \geq 0, \;\; i \in N_m \right\}.$$

Pasul 1. Rezolvăm problema

$$\min \{ f_s(x) \mid x \in X_k \} \tag{8}$$

folosind algoritmul simplex dual.

Alegem $x^{k+1} = \text{argmin } \{ f_s(x) \mid x \in X_k \}$.

Dacă $X \subseteq X_k$, $x^{k+1} \in X$ şi x^{k+1} este sigura soluţie optimă din mulţimea soluţiilor admise X, atunci

$$x^{k+1} = \text{argmin}^L \{ F(x) \mid x \in X \},$$

şi problema $Z_L(F, X)$ este rezolvată.

Pasul 2. Dacă $x^{k+1} \in X$ şi x^{k+1} nu este singura soluţie optimă din mulţimea X, atunci:

$$\overline{f_s} = f_s(x^{k+1})$$

$$X_{k+1} = \left\{ x \in X_k \mid f_i(x) = \overline{f_i}, \; i = \overline{1, s} \right\}, s = s + 1$$

Sari la Pasul 1.

Dacă $x^{k+1} \notin X$, atunci sari la Pasul 3.

Pasul 3. Vom defini mulţimea I_{k+1} de indici - indicii pentru care constrângerile sunt încălcate în punctul x^{k+1} - cu ajutorul căreia vom crea noi constrângeri pentru problema $Z_L(F, X)$ astfel:

$$I_{k+1} = \left\{ i \mid g_i(x^{k+1}) > 0 \right\}.$$

În continuare vom construi mulţimea poliedrală X_{k+1} adăugând mulţimii X_k constrângerile date de inegalitatea

$$\langle \nabla g_i(x^{k+1}), x - x^{k+1} \rangle + g_i(x^{k+1}) \leq 0,$$

$$\text{cu } i \in N_{k+1} = \left\{ j \in I_{k+1} \mid g_j(x^{k+1}) = \min_{i \in I_{k+1}} g_i(x^{k+1}) \right\}.$$

Astfel obţinem mulţimea

$$X_{k+1} = \left\{ x \in X_k \mid \langle \nabla g_i(x^{k+1}), x - x^{k+1} \rangle + g_i(x^{k+1}) \leq 0, i \in N_{k+1} \right\}.$$

Incrementăm k : k = k + 1 şi revenim la Pasul 1.

Pentru rezolvarea subproblemelor de tipul 8 este recomandată folosirea algoritmului simplex dual, care permite utilizarea soluţiei obţinute anterior ca bază pentru noul domeniu obţinut după adăugarea de constrângeri.

Convergenţa algoritmului prezentat anterior este asigurată de următoarea teoremă:

TEOREMA 6. *Dacă funcţiile $g_i : \mathbb{R}^n \to \mathbb{R}, i \in N_m$ sunt convexe şi diferenţiabile continuu, iar problema $Z_L(F, X)$ are o soluţie optimă finită, atunci mulţimea punctelor generate de algoritmul prezentat anterior converge către soluţia optimă lexicografic a problemei $Z_L(F, X)$.*

Demonstraţie. Pornind de la ideea că problema $Z_L(F, X)$ are o soluţie optimă lexicografic finită, deducem că de la un anumit număr p_0, secvenţa de puncte este inclusă într-o mulţime mărginită $\{x^p\}$.

Fie $\{x^k\}$ o submulţime a mulţimii $\{x^p\}$ care converge la un punct x^*.

Vom considera submulţimea $\{x^t\}$ de puncte generate de condiţia i din relaţia $\langle \nabla g_i(x^j), x - x^j \rangle + g_i(x^j) \leq 0$.

Dacă la fiecare iteraţie hiperplanul este tăiat prin aplicarea unei noi constrângeri (cea mai puternică sau cea mai încălcată constrângere), atunci de la un $k \geq k_0$, restricţia $g_i(x^k) \leq 0$ este respectată, adică x^k aparţine mulţimii soluţiilor admise ale problemei sau submulţimea $\{x^t\}$ este infinită.

Vom considera cazul în care submulţimea $\{x^p\}$ este infinită, astfel că pentru orice $t' > t$, are loc:

$$\langle \nabla g_i(x^t), x^{t'} - x^t \rangle + g_i(x^t) \leq 0.$$

Aplicând inegalitatea Cauchy-Bunyakovsky obţinem:

$$g_i(x^t) \leq \left\| \nabla g_i(x^t) \right\| \cdot \left\| x^{t'} - x^t \right\|$$

Considerând $\left\| x^{t'} - x^t \right\| \to 0$ şi $\left\| \nabla g_i(x^t) \right\| \to \| \nabla g_i(x^*) \|$, din ultima inegalitate rezultă că

$$\Rightarrow \left\| g_i(x^t) \right\| \to \| g_i(x^*) \| \leq 0,$$

de unde deducem că x^* este o soluţie admisă a problemei $Z_L(F, X)$.

Pe de altă parte, dacă $\overline{x}$ este o soluţie optimă a problemei $Z_L(F, X)$, atunci inegalitatea $F(x^t) \geq^L F(\overline{x})$ are loc pentru fiecare iteraţie a algoritmului, prin trecerea la limită obţinem că $F(x^*) \geq^L F(\overline{x})$. Deci x^* este soluţia optimă lexicografic a problemei $Z_L(F, X)$, iar teorema este demonstrată.

□

În metoda propusă, mulţimea $\{x^k\}$ se construieşte pornind de la puncte x^k care nu sunt soluţii pentru problema originală. Altfel, procesul de calcul nu se va opri decât la valori mari ale lui s, şi doar când găsim o soluţie admisă. Convergenţa la o soluţie optimă lexicografic este garantată prin acest algoritm dacă mulţimea soluţiilor posibile este convexă.

EXEMPLUL 5. *Fie $F : \mathbb{R}^2 \to \mathbb{R}^2, F(x) = (x_1 + 2x_2, -x_1 + 3x_2 + 5)$ şi fie $g_1, g_2, g_3 : \mathbb{R}^2 \to \mathbb{R}$ funcţii convexe, constrângeri pentru problema noastră,*

$$g_1(x) = x_1 - x_2 - 3 \leq 0,\ g_2(x) = -x_1 + 2 \leq 0,\ g_3(x) = -x_2 + 2 \leq 0.$$

Găsiţi soluţia optimă lexicografic pentru minimizarea funcţiei F, ţinând cont de constrângerile date.

SOLUŢIA 5. *Vom rescrie problema astfel:*

$$(P): \quad \begin{cases} F(x) = (f_1(x), f_2(x)) = (x_1 + 2x_2, -x_1 + 3x_2 + 5) \to \min \\ g_1(x) = x_1 - x_2 - 3 \leq 0 \\ g_2(x) = -x_1 + 2 \leq 0 \\ g_3(x) = -x_2 + 2 \leq 0 \\ x_1, x_2 \geq 0 \end{cases}$$

$$X = \left\{x \in \mathbb{R}^2 \mid g_1(x) \leq 0,\ g_2(x) \leq 0,\ g_3(x) \leq 0,\ x \geq 0_2\right\}$$
$$= \left\{x = (x_1, x_2) \in \mathbb{R}^2 \mid x_1 - x_2 \leq 3,\ -x_1 \leq -2,\ -x_2 \leq -2,\ x_1, x_2 \geq 0\right\}$$

Fie $s = 1$ *şi* $k = 0$. *Fie* $x^0 = (0,0)$. *Calculăm prima mulţime poliedrală* X_0 :

$$X_0 = \left\{x \in \mathbb{R}^2 \mid \langle \nabla g_i(x^0), x - x^0 \rangle + g_i(x^0) \leq 0, i = \overline{1,3}\right\}$$

$$\nabla g_1(x) = \left(\frac{\partial g_1}{\partial x_1}, \frac{\partial g_1}{\partial x_2}\right) = (1, -1)$$
$$\nabla g_2(x) = \left(\frac{\partial g_2}{\partial x_1}, \frac{\partial g_2}{\partial x_2}\right) = (-1, 0) \quad , \text{ pentru } \forall x = (x_1, x_2) \in \mathbb{R}^2$$
$$\nabla g_3(x) = \left(\frac{\partial g_3}{\partial x_1}, \frac{\partial g_3}{\partial x_2}\right) = (0, -1)$$

Calculăm valorile funcţiilor g_i *în* $x^0 = (0,0)$*:*

$$g_1(0,0) = 0 - 0 - 3 = -3$$
$$g_2(0,0) = -0 + 2 = 2$$
$$g_3(0,0) = -0 + 2 = 2$$

$$\langle \nabla g_1(0,0), (x_1, x_2) - (0,0) \rangle + g_1(0,0) \leq 0 \Leftrightarrow$$
$$\Leftrightarrow \langle (1,-1), (x1, x2) \rangle - 3 \leq 0 \Leftrightarrow$$
$$\Leftrightarrow x_1 - x_2 \leq 3 \textit{(*)}$$

$$\langle \nabla g_2(0,0), (x_1, x_2) - (0,0) \rangle + g_2(0,0) \leq 0 \Leftrightarrow$$
$$\Leftrightarrow \langle (-1,0), (x1, x2) \rangle + 2 \leq 0 \Leftrightarrow$$
$$\Leftrightarrow -x_1 \leq -2 \textit{(**)}$$

$$\langle \nabla g_3(0,0), (x_1, x_2) - (0,0) \rangle + g_3(0,0) \leq 0 \Leftrightarrow$$
$$\Leftrightarrow \langle (0,-1), (x1, x2) \rangle + 2 \leq 0 \Leftrightarrow$$
$$\Leftrightarrow -x_2 \leq -2 \textit{(***)}$$

$$x_1, x_2 \geq 0 \textit{(****)}$$

Din (*), (**), (***) şi (****) $\Rightarrow$
$\Rightarrow X_0 = \{(x_1, x_2) \in \mathbb{R}^2 \mid x_1 - x_2 \leq 3,\ -x_1 \leq -2,\ -x_2 \leq -2,\ x_1 \geq 0,\ x_2 \geq 0\}$

Pasul 1: $\min\{f_1(x) \mid x \in X_0\}, \quad f_1(x) = x_1 + 2x_2$

$$(P1): \begin{cases} f_1(x) = x_1 + 2x_2 \to \min \\ x_1 - x_2 \leq 3 \\ -x_1 \leq -2 \\ -x_2 \leq -2 \\ x_1, x_2 \geq 0 \end{cases} \quad \to \quad (P2): \begin{cases} f_1(x) = x_1 + 2x_2 \to \min \\ x_1 - x_2 + x_3 = 3 \\ -x_1 + x_4 = -2 \\ -x_2 + x_5 = -2 \\ x_1, x_2, x_3, x_4, x_5 \geq 0 \end{cases}$$

Vom rezolva problema (P2) folosind algoritmul simplex:

$$A = \begin{pmatrix} 1 & -1 & 1 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 1 \end{pmatrix} = (A^1 A^2 A^3 A^4 A^5),\ b = \begin{pmatrix} 3 \\ -2 \\ -2 \end{pmatrix},\ c = (1, 2, 0, 0, 0)$$

Alegem baza $B = (A^3 A^4 A^5)$ *şi construim tabelul:*

Tabelul 1	A^3	A^4	A^5	*Test b.d.a.*
A^1	$\alpha_{1,3} = 1$	$\boxed{\alpha_{1,4} = -1}$	$\alpha_{1,5} = 0$	$\alpha_{1,0} = -1$
A^2	$\alpha_{2,3} = -1$	$\alpha_{2,4} = 0$	$\alpha_{2,5} = -1$	$\alpha_{2,0} = -2$
Test b.p.a.	$\alpha_{0,3} = 3$	$\alpha_{0,4} = -2$	$\alpha_{0,5} = -2$	$\alpha_{0,0} = 0$

Elementele din sectorul vertical marcat sunt toate ≤ 0*, deci avem o baza dual admisibilă, însă în cel orizontal nu sunt toate* ≥ 0 *(*$\alpha_{0,4}, \alpha_{0,5} < 0$*), deci baza noastră nu este şi primal admisibilă. Vom aplica algoritmul simplex dual.*
Alegem ca pivot elementul $\alpha_{1,4} = -1$ *şi construim noul tabel:*

Tabelul 2	A^3	A^1	A^5	*Test b.d.a.*
A^4	$\alpha_{4,3} = 1$	$\alpha_{4,1} = -1$	$\alpha_{4,5} = 0$	$\alpha_{4,0} = -1$
A^2	$\alpha_{2,3} = -1$	$\alpha_{2,1} = 0$	$\boxed{\alpha_{2,5} = -1}$	$\alpha_{2,0} = -2$
Test b.p.a.	$\alpha_{0,3} = 1$	$\alpha_{0,1} = 2$	$\alpha_{0,5} = -2$	$\alpha_{0,0} = 2$

Şi în acest caz avem o bază dual admisibilă, dar înca avem elemente strict negative în sectorul orizontal marcat, deci vom continua algoritmul.
Alegem ca pivot elementul $\alpha_{2,5} = -1$ *şi construim noul tabel:*

Tabelul 3	A^3	A^1	A^2	*Test b.d.a.*
A^4	$\alpha_{4,3} = 1$	$\alpha_{4,1} = -1$	$\alpha_{4,2} = 0$	$\alpha_{4,0} = -1$
A^5	$\alpha_{5,3} = -1$	$\alpha_{5,1} = 0$	$\alpha_{5,2} = -1$	$\alpha_{5,0} = -2$
Test b.p.a.	$\alpha_{0,3} = 3$	$\alpha_{0,1} = 2$	$\alpha_{0,2} = 2$	$\alpha_{0,0} = 6$

Toate elementele din sectorul vertical marcat sunt negative (≤ 0), deci avem bază dual admisibilă şi toate elementele din sectorul orizontal marcat sunt pozitive (≥ 0), deci avem bază primal admisibilă. Din aceasta deducem că ($A^3A^1A^2$) este o bază optimă pentru (P2), valoarea minimă fiind $\alpha_{0,0} = 6$, iar soluţia optimă $x' = (2, 2, 3, 0, 0)$.

Pentru problema (P1) soluţia optimă este $x^1 = (2, 2)$.

Deoarece $X = X_0$, iar $x^1 \in X$ şi este singura soluţie optimă din această mulţime, $x^1 = (2, 2)$ este soluţia optimă lexicografic a problemei iniţiale.

OBSERVAŢIA 3. *Dacă am fi ales $x^0 = (0, 1)$, am fi ajuns la acelaşi rezultat pe care l-am obţinut şi pornind de la $x^0 = (0, 0)$:*

$$X_0 = \left\{x \in \mathbb{R}^2 \mid \langle \nabla g_i(x^0), x - x^0\rangle + g_i(x^0) \leq 0, i = \overline{1,3}\right\}$$

$$\nabla g_1(x) = (1, -1), \quad \forall x = (x_1, x_2) \in \mathbb{R}^2$$
$$\nabla g_2(x) = (-1, 0), \quad \forall x = (x_1, x_2) \in \mathbb{R}^2$$
$$\nabla g_3(x) = (0, -1), \quad \forall x = (x_1, x_2) \in \mathbb{R}^2.$$

Calculăm valorile funcţiilor g_i în $x^0 = (0, 1)$:

$$g_1(0, 1) = 0 - 1 - 3 = -4$$
$$g_2(0, 1) = -0 + 2 = 2$$
$$g_1(0, 1) = -1 + 2 = 1$$

$$\langle \nabla g_1(0, 1), (x_1, x_2) - (0, 1)\rangle + g_1(0, 0) \leq 0 \Leftrightarrow$$
$$\Leftrightarrow \langle (1, -1), (x_1, x_2 - 1)\rangle - 4 \leq 0 \Leftrightarrow$$
$$\Leftrightarrow x_1 - x_2 \leq 3 \quad (*)$$

$$\langle \nabla g_2(0, 1), (x_1, x_2) - (0, 1)\rangle + g_2(0, 0) \leq 0 \Leftrightarrow$$
$$\Leftrightarrow \langle (-1, 0), (x_1, x_2 - 1)\rangle + 2 \leq 0 \Leftrightarrow$$
$$\Leftrightarrow -x_1 \leq -2 \quad (**)$$

$$\langle \nabla g_3(0, 1), (x_1, x_2) - (0, 1)\rangle + g_3(0, 0) \leq 0 \Leftrightarrow$$
$$\Leftrightarrow \langle (0, -1), (x_1, x_2 - 1)\rangle + 1 \leq 0 \Leftrightarrow$$
$$\Leftrightarrow -x_2 \leq -2 \quad (***)$$

$$x_1, x_2 \geq 0 \quad (****)$$

Din aceste calcule observăm că obţinem pentru X_0 aceiaşi mulţime cum am obţinut folosind $x_0 = (0, 0)$.

2.6. Alegerea soluţiei optime lexicografic folosind scalarizarea .

Reamintim faptul că optimizarea lexicografică în minimizarea unei funcţii scop

$$F(x) = (f_1(x), f_2(x), ..., f_\ell(x)), \ell \geq 2,$$

unde $f_i(x)$, $i = \overline{1, \ell}$, $x \in X$ (X fiind mulţimea de soluţii admisibile) se numesc criterii, urmăreşte găsirea unei soluţii optime lexicografic, pe fondul realizării constrăngerii ca aceste criterii să fie respectate în ordinea dată de importanţa lor, criteriul cel mai important fiind $f_1(x)$, iar $f_\ell(x)$ cel mai puţin important. Scalarizarea este o metodă prin care un vector este transformat într-o valoare reală folosind diferite tehnici, cum ar fi tehnica sumelor ponderate, funcţiile de calculat distanţe, minimul, maximul, etc. Aceaste abordări presupun gestionearea importanţelor criteriilor stabilite, cu scopul de a minimiza valoarea funcţiei de scop.

O abordare intuitivă a scalarizării în alegerea soluţiei optime lexicografic este aplicarea metodei sumelor ponderate, prin care se atribuie fiecărui criteriu o pondere proporţională cu importanţa sa.

Astfel, vom atribui fiecărui obiectiv o pondere $w_i > 0$, apoi se va calcula:

$$val_{min} = \min \left\{ \sum_{i=1}^{\ell} w_i f_i(x) \mid x \in X \right\},$$

iar mulţimea optimelor lexicografic va fi

$$\left\{ x \in X \mid \sum_{i=1}^{\ell} w_i f_i(x) = val_{min} \right\}.$$

Observaţia 4. *Alegerea ponderilor şi elementele vectorilor sunt aspecte esenţiale în aplicarea scalarizării prin sume ponderate:*

- *Pot exista criterii care au aceiaşi pondere $\Leftrightarrow$ sunt la fel de importante.*
- *Dacă vectorii sunt formaţi atât din elemente negative, cât şi din elemente pozitive, trebuie să adaptăm corespunzător ponderile pentru ca abordarea prezentată anterior să funcţioneaze. În general, cu cât diferenţele $x_{i+1} - x_i$ sunt mai mari, metoda sumelor ponderate oferă rezultate mai corecte.*

Exemplul 6. *Fie $X \subseteq \mathbb{R}^4$, $Y \subseteq \mathbb{R}^3$ mulţimi finite ce conţin următoarele elemente:*

$$X = \{x_1 = (-1, 0, 0, 1), x_2 = (-1, -1, 3, 0), x_3 = (-1, 2, -3, -4)\},$$
$$Y = \{y_1 = (1, 0, 2), y_2 = (1, 0, 0), y_3 = (1, 1, 2), y_4 = (1, 1, 0)\}.$$

Sortaţi elementele mulţimilor X şi Y ordonând lexicografic vectorii, apoi sortaţile în funcţie de valoarea lor obţinută prin scalarizare. Care este elementul optim în fiecare caz pentru problema de minimizare?

Soluţia 6. ***Pentru X:***
Sortarea folosind ordonarea lexicografică:

$$x_2 = (-1,-1,3,0) \leq x_1 = (-1,0,0,1) \leq x_3 = (-1,2,-3,-4)$$
$$\Rightarrow x_2 - x_1 - x_3$$
$$\Rightarrow \textit{ Elementul optim este } x_2.$$

Sortarea folosind valoarea obţinută prin scalarizare:

- Alegem vectorul de ponderi $w = (4,3,2,1)$ *şi calculăm sumele ponderate pentru fiecare:*

$s_{x_1} = (-1)\cdot 4 + 0\cdot 3 + 0\cdot 2 + 1\cdot 1 = -4 + 1 = -3$
$s_{x_2} = (-1)\cdot 4 + (-1)\cdot 3 + 3\cdot 2 + 0\cdot 1 = -4 - 3 + 6 = -1$
$s_{x_3} = (-1)\cdot 4 + 2\cdot 3 + (-3)\cdot 2 + (-4)\cdot 1 = -4 + 6 - 6 - 4 = -8$

$$\Rightarrow x_3 - x_1 - x_2$$
$$\Rightarrow \textit{ Elementul optim este } x_3.$$

Rezultatul pe care l-am obţinut ne arată că ponderile noastre nu au fost alese destul de reprezentativ pentru a obţine în mod corect sortarea şi valoarea optimă.

- Alegem vectorul de ponderi $w = (30,20,5,1)$ *şi calculăm sumele ponderate pentru fiecare:*

$s_{x_1} = (-1)\cdot 30 + 0\cdot 20 + 0\cdot 5 + 1\cdot 1 = -30 + 1 = -29$
$s_{x_2} = (-1)\cdot 30 + (-1)\cdot 20 + 3\cdot 5 + 0\cdot 1 = -30 - 20 + 15 = -35$
$s_{x_3} = (-1)\cdot 30 + 2\cdot 20 + (-3)\cdot 5 + (-4)\cdot 1 = -30 + 40 - 15 - 4 = -9$

$$\Rightarrow x_2 - x_1 - x_3$$
$$\Rightarrow \textit{ Elementul optim este } x_2, \textit{ iar sortarea este aceeaşi cu cea obţinută folosind ordonarea lexicografică.}$$

Pentru Y:

Sortarea folosind ordonarea lexicografică:

$$y_2 = (1,0,0) \leq y_1 = (1,0,2) \leq y_4 = (1,1,0) \leq y_3 = (1,1,2)$$
$$\Rightarrow y_2 - y_1 - y_4 - y_3$$
$$\Rightarrow \textit{ Elementul optim este } y_2.$$

Sortarea folosind valoarea obţinută prin scalarizare:

- Alegem vectorul de ponderi $w = (3,2,1)$ *şi calculăm sumele ponderate pentru fiecare:*

$s_{y_1} = 1\cdot 3 + 0\cdot 2 + 2\cdot 1 = 3 + 2 = 5$
$s_{y_2} = 1\cdot 3 + 0\cdot 2 + 0\cdot 1 = 3$
$s_{y_3} = 1\cdot 3 + 1\cdot 2 + 2\cdot 1 = 3 + 2 + 2 = 7$
$s_{y_4} = 1\cdot 3 + 1\cdot 2 + 0\cdot 1 = 3 + 2 = 5$

$$\Rightarrow y_2 - y_1 - y_4 - y_3 \textit{ sau } y_2 - y_4 - y_1 - y_3$$
$$\Rightarrow \textit{ Elementul optim este } y_2.$$

Am obţinut acelaşi optim, însă sortările nu coincid (y_1 *şi* y_4 *sunt pe aceaşi nivel). Vom încerca să ajustăm ponderile ca să ajungem exact la aceiaşi ordine în sortări.*

- Alegem vectorul de ponderi $w = (20,10,1)$ *şi calculăm sumele ponderate pentru fiecare:*

$s_{y_1} = 1\cdot 20 + 0\cdot 10 + 2\cdot 1 = 20 + 2 = 22$

$s_{y_2} = 1 \cdot 20 + 0 \cdot 10 + 0 \cdot 1 = 20$
$s_{y_3} = 1 \cdot 20 + 1 \cdot 10 + 2 \cdot 1 = 20 + 10 + 2 = 32$
$s_{y_4} = 1 \cdot 20 + 1 \cdot 10 + 0 \cdot 1 = 20 + 10 = 30$

$$\Rightarrow y_2 - y_1 - y_4 - y_3$$

$\Rightarrow$ *Elementul optim este* y_2*, iar sortările coincid.*

În capitolul următor vom folosi principiile algoritmilor şi metodelor pe care le-am prezentat anterior în acest capitol, pentru a oferi o privire de ansamblu asupra a ceea ce presupune optimizarea lexicografică în planificarea tratamentului cu radioterapie cu intensitate modulată pentru bolnavii de cancer.

3. METODA ORDONĂRII LEXICOGRAFICE ÎN PLANIFICAREA IMRT

Radioterapia cu intensitate modulată (IMRT) este un tip avansat de radioterapie, utilizat pentru tratarea cancerului şi a tumorilor benigne. Planificarea acestor tratamente poate fi privită ca o problemă de optimizare vectorială, deoarece implică mai multe obiective de planificare, ce urmăresc adaptarea distribuţiei dozei terapeutice la geometria masei tumorale, vizând protejarea în mod optim, a ţesututilor sănătoase din vecinătate. Pentru această provocare, vom aborda strategia de optimizare a ordonării lexicografice (LO) şi vom demonstra eficienţa utilizării acesteia.

Materialul prezentat în acestă secţiune a fost documentat în principal din articolul [4], dar şi din următoarele referinţe bibliografice: [1], [2] şi [3].

3.1. Planificarea inversă în IMRT .

Radioterapia cu Intensitate Modulată, pe scurt, IMRT, este un tip de tratament în care, utilizând informaţiile anatomice tridimensionale ale tumorii şi ale ţesuturilor sănătoase înconjurătoare, se modulează intensitatea fasciculelor de radiaţii (câmpul de radiaţie se împarte în fascicule mici numite beamlets), permiţând o dozare mai mare asupra tumorii şi minimizând expunerea ţesuturilor înconjurătoare la radiaţii. Planificarea dozării se face, în cele mai multe cazuri, folosind tehnica planificării inverse, care are trei etape:

(1) Alegerea structurilor anatomice care vor fi utilizate în planul de optimizare;
(2) Distribuţia unei doze iniţiale de radiaţii (prin fascicule) în câmpul de iradiere;
(3) Optimizarea fluenţelor fasciculului, determinate de un obiectiv sau de o funcţie de cost.

Tehnica planificării inverse a dus deseori la planificări IMRT foarte eficiente însă, în practică, găsirea soluţiei optime se atinge în urma încercărilor multiple, deoarece găsirea funcţiei de cost, ajustarea ei şi a parametrilor ei astfel încât să ajungem la o soluţie mai bună se face printr-o abordare încercare -

eroare (fiecare caz are particularităţile lui). De asemenea, conflictul dintre necesitatea creşterii dozei de radiaţie la tumoare şi reducerea dozei la ţesuturile sănătoase poate influienţa eficienţa soluţiei utilizate până atunci, necesitând o nouă abordare după un anumit timp.

În planificarea inversă, datorită faptului că fluenţele fasciculelor de radiaţii sunt specifice pentru fiecare structură anatomică aleasă pentru plan, se creează mai multe obiective de luat în considerare pentru planul de optimizare. Aceste obiective au dus la ideea utilizării strategiilor de optimizare vectorială.

3.2. Optimizarea lexicografică în planificarea inversă .

Optimizarea lexicografică este o strategie de optimizare vectorială, care s-a dovedit a fi eficientă în planificarea IMRT. Ea are două etape principale: ordonarea lexicografică a obiectivelor (gruparea şi ierarhizarea lor pe nivele de importanţă) şi optimizarea efectivă. Acest tip de optimizare se asigură la fiecare pas că obiectivul mai important este satisfăcut înainte de a se lua în considerare următorul obiectiv din ierarhie, aspect esenţial in planificarea IMRT.

3.2.1. *Modelarea matematică a problemei.*
În primul rând, vom avea nevoie de o funcţie scop $F(x)$ pe care dorim să o minimizăm. Aceasta funcţie $F(x)$ va fi o colecţie de funcţii individuale $f_i(x)$ care vor corespunde fiecărui criteriu de planificare. Transcrisă în limbaj matematic, problema noastră de optimizare ar arăta în felul următor:

Să se găsească $x \in \mathbb{R}^n$ care minimizează funcţia $F(x)$, ţinând cont de constrângerile date:

$$\begin{cases} F(x) = (f_1(x), f_2(x), \ldots, f_\ell(x)) \to \min \\ g_j(x) \leq 0, \quad j = 1, 2, \ldots, m \\ h_k(x) = 0, \quad k = 1, 2, \ldots, e \end{cases}$$

unde $x \in \mathbb{R}^n$ este vectorul ce reprezintă variabilele de proiectare (în cazul nostru, variabilele sunt intensităţile fasciculelor de radiaţii);
$f_i(x)$ criteriul de planificare (funcţia obiectiv) cu numărul i;
ℓ = numărul de funcţii obiectiv;
m = numărul de constrângeri sub formă de inecuaţii;
e = numărul de constrângeri sub formă de ecuaţii.

În teorie, soluţia cea mai bună ar fi cea în care funcţiile $f_i(x)$ sunt minimizate simultan, dar în practică s-a observat că, de obicei, nu se poate găsi o astfel de soluţie ideală. Pentru a depăşi acest obstacol, problema noastră devine aceea de a găsi cea mai bună soluţie de compromis, care să ţină cont de preferinţele celui care decide planul care se va folosi în tratament.

O metodă care simplifică şi eficientizează problema găsirii planului optim al tratametului cu IMRT este scalarizarea lui F prin combinarea funcţiilor

obiectiv f_i, folosind metoda sumei ponderate:

$$\min f_{\text{total}}(x) = \sum_{i=1}^{N} w_i f_i(x), \text{ unde} \tag{9}$$

w_i sunt ponderi atribuite fiecărei funcţii obiectiv, în funcţie de priorităţile stabilite de planificator

Optimizarea lexicografică (LO) permite încorporarea acestor priorităţi direct în procesul de optimizare, prin crearea de constrângeri (LO este adesea menţionată ca o metodă de reducere a spaţiului fezabil), fără a mai fi necesară utilizarea unei funcţii de cost. Matematic, aceasta s-ar scrie în felul următor:

$$\begin{cases} \min f_i(x) \\ f_j(x) \leq f_j(x_j^*) \end{cases}$$

$$\text{unde } i = 1, 2, \ldots, \ell;\ j = 1, 2, \ldots, i-1 \text{ dacă } i > 1, \text{ iar}$$
$$x_j^* = \operatorname{argmin} f_j(x).$$

Pasul 1.

Medicul va alcătui o ierarhie a obiectivelor în funcţie de importanţa lor, nivelul cu cea mai mare prioritate fiind $i = 1$, iar $i = \ell$ ultimul nivel de importanţă. Într-o astfel de ierarhie, pot exista mai multe obiective cu aceiaşi prioritate.

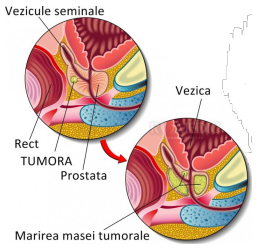
Pasul 2.

Se utilizează un algoritm de căutare care va rezolva pe rând fiecare nivel de optimizare. În timp ce algoritmul progresează de la nivelul 1 la nivelul ℓ, funcţiile obiectiv precedente vor fi transformate în constrângeri de tip inegalităţi, cu valori limită $f_j(x_j^*)$, obţinute ca soluţii a priori la $x_j^* = \operatorname{argmin} f_j(X)$, care respectă constrângerile de la nivelele superioare. Dacă există constrângeri cu acelaşi nivel de prioritate, acestea pot fi combinate folosind o metodă similară cu (9). Astfel, metoda ordonării lexicografice reduce gradual spaţiul, adăugând constrângeri cu fiecare nivel rezolvat.

În cele ce urmează, vom prezenta o abordare LO în planificarea IMRT pentru un caz relativ simplu de prostată, pentru a înţelege comportamentul de bază al metodei ordonării lexicografice în găsirea planului optim de tratament.

3.2.2. *Cazul IMRT de prostată.*

Cancerul de prostată este una dintre cele mai întâlnite tipuri de cancer la bărbaţi. Datorită localizării ei într-o zonă mai izolată, a sensibilităţii la radiaţii a ţesuturilor şi a toleranţei la doza de radiaţie a structurilor anatomice înconjurătoare, această



afecţiune a fost mult studiată şi i s-
au găsit numeroase metode de tratament, radioterapia cu intensitate modulată ocupând un loc de cinste.

Planificarea inversă

Utilizând etapele din metoda planificării inverse, în primul rând se stabilesc structurile anatomice care vor fi utilizate în planul de optimizare:
- volumul ţintă: prostata
- structurile critice: vezica urinară şi rectul (vom studia o abordare simplificată a cazului)

Folosind imaginile obţinute prin tomografia computerizată(CT) şi prin rezonanţă magnetică (RMN), se definesc contururi pentru structurile normale şi ţinte în corpul pacientului. Aceste structuri vor fi înconjurate de patru câmpuri de 6 MV (energia de 6 MV - milioane de electronvolţi - este des întâlnită pentru tratamentele cu radiaţii în oncologie; reprezintă energia fasciculelor de radiaţie), subdivizate în aproximativ 570 fascicule, având fiecare dimensiunea de 5×5 mm^2.

Pentru a stabili distribuţia dozei iniţiale de radiaţii, vom defini întâi obiectivele de planificare, ordonându-le după impotanţă (în funcţie de preferinţele medicului), în 4 nivele:

Tabela 1 – Planul A

Nivelul 1	Iradierea uniformă la volumul ţintă planificat al prostatei, cu variaţii de doză de maximum +/- 5%
Nivelul 2	Creşterea dozei medii primite de volumul ţintă la 90 Gy(Grey)
Nivelul 3	Minimizarea dozei de iradiere de la nivelul rectului utilizând cinci criterii convenţionale de doză-volum
Nivelul 4	Minimizarea dozei totale de iradiere primite de vezica urinară

Metoda optimizării lexicografice în practică

Rezultatele prezentate în această subsecţiune sunt preluate dintr-un articol scris de Kyung-Wook Jee, Daniel L McSham şi Benedick A Frasas de la Departamentul de Radiaţii în Oncologie al Universităţii din Michigan, şi reprezintă soluţii obţinute folosind un sistem de optimizare dezvoltat de ei şi folosit în practică. Acest sistem funcţionează într-un context destul de general al planificării inverse în IMRT, deoarece foloseşte mai multe tipuri de funcţii de cost, diverşi algoritmi de căutare eficienţi, algoritmi de calculare a dozei de iradiere prin convoluţie folosind matrici de cuantificare a contribuţiilor la doză, metode flexibile de calculare a Jacobienilor pentru funcţiile obiectiv şi constrângeri,

metrici de planificare, toate acestea fiind elemente esenţiale pentru a obţine o planificare optimă.

Utilizând acest sistem propriu, parcurgând cele patru nivele (funcţii obiectiv), histogramele doză-volum (DVH) pentru cele 3 structuri ţintă, s-au modificat după cum se poate observa în Figura 3.1 de mai jos:

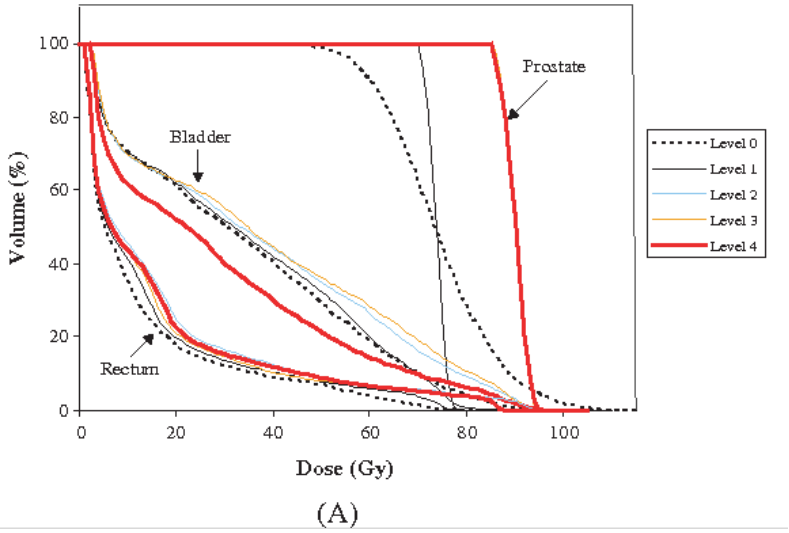


Figura. 3.1 – Histogramele doză-volum pentru volumele ţintă planificate (rect, vezică urinară şi prostată), pe cele 4 nivele de optimizare

Rezolvarea fiecărui nivel de optimizare s-a realizat după cum urmează:

Nivelul 0

Pentru început, fiecărei structuri anatomice i s-a atribuit o doză de fascicule cu intensităţi stabilite random (distribuţia se poate vedea prin linia punctată).

Nivelul 1

La acest nivel, obiectivul optimizării este să asigure o acoperire uniformă şi consistentă a prosatei cu doza de radiaţie, cu o variaţie de +/- 5% faţă de doza medie (constrângere asupra variaţiei DVH pentru PTV - volumul ţintă planificat). Având în vedere ca obiectivul a vizat exclusiv volumul ţintă planificat (prostata), obiectivul s-a realizat rapid şi nu a afectat seminificativ DVH-ul celorlalte structuri.

Nivelul 2

Se continuă cu al doilea obiectiv de a creşte doza medie pentru PTV la 90 Gy, şi se observă că s-a pastrat constrângerea de uniformitate obţinută la

nivelul anterior. Din nou obiectivul nu vizează celelalte două structuri, deci micile modificări din DVH-urile lor nu sunt foarte relevante.

Nivelul 3

Cele cinci criterii convenţionale pentru iradierea de la nivelul rectului sunt: volumul rectal ce primeşte o doză de radiaţie mai mare de 65 Gy, 70 Gy, 75 Gy, 80 Gy, 85.2 Gy să fie mai mic de 50%, 35%, 25%, 15%, 0% din volumul total. După realizarea acestui obiectiv (cu excepţia criteriului 85.2 Gy 0%, pentru care se poate observa doar o îmbunătăţire - volumul care primeşte exact sau mai mult de 85.2 a scăzut de la 3.4% la 2.8%), urmărind respectarea constrângerilor obţinute anterior, s-au creat cele cinci noi constrângeri pentru dozarea de la nivelul rectului.

Nivelul 4

La ultimul nivel se urmăreşte minimizarea dozei de radiaţie de la nivelul vezicii urinare, şi se observă că în urma optimizării, doza medie a scăzut de la 37.9 Gy la 28.6 Gy.

Astfel, ţinând cont de constrângerile obţinute la fiecare nivel de optimizare, s-a obţinut soluţia finală de planificare.

3.2.3. *O implementare a sistemului de optimizare.*

În cele ce urmează vom prezenta o implementare simplificată a sistemului de optimizare descris anterior (dezvoltată în limbajul Python), care utilizează principii ale metodei tăierii planului şi ale metodei scalarizării.

Elementele de bază ale algoritmului

- x - un vector cu 570 de elemente (corespunzător numărului de fascicule) ce reprezintă intensităţile fasciculelor de radiaţii (valori pozitive). În practică, aceste fascicule sunt împărţite în 4 câmpuri, dar pentru a înţelege mai uşor ideea din spate a algoritmului şi pentru a eficentiza calculele, noi vom împărţi fasciculele în 3 secţiuni. Vom considera că elementele lui x (fasciculele) sunt reordonate astfel încât cele de pe poziţiile de la 0 - 269 sunt orientate spre prostata, de la 270 - 419 sunt orientate spre rect, iar de la 420 - 569 sunt orientate spre vezica urinară.

$$x = (\underbrace{x_0, \ldots, x_{269}}_{\text{prostată}}, \underbrace{x_{270}, \ldots, x_{419}}_{\text{rect}}, \underbrace{x_{420}, \ldots, x_{569}}_{\text{vezica urinară}})$$

  Vom considera că pentru fiecare x_i, cu cât indicele i e mai mic în fiecare secţiune, cu atat fasciculul e poziţionat mai aproape de tumoare.
- medie_p = media intensităţilor fasciculelor centrate pe prostată;
- medie_r = media intensităţilor fasciculelor centrate pe rect;
- medie_v = media intensităţilor fasciculelor centrate pe vezica urinară;

- w = ponderile pentru fiecare dintre cele 3 medii: w =($w_{\text{prostată}}$, w_{rect}, $w_{\text{vezică}}$), ce se vor folosi pentru calcularea valorilor funcţiei scop;
- $n_i = scalarizare(x_i, w), i = \overline{1,4}$, sunt funcţiile obiectiv corespunzătoare fiecărui nivel de optimizare (se vor calcula prin scalarizare, folosind mediile celor 3 structuri şi vectorul de ponderi dat), iar $F(x) = (n_1(x), n_2(x), n_3(x), n_4(x))$ este funcţia scop, pe care o avem de minimizat.

Implementarea efectivă

Pentru început avem nevoie de o funcţie pe care o vom numi **nivel**_0(x). Aceasta ne va iniţializa variabila x (intensităţile fasciculelor) cu valori descrescătoare din intervalul $[0, 100]$, în mod uniform, dar random, separat pentru fiecare secţiune ce reprezintă structurile prostată, rect şi vezică urinară (distribuţia iniţială va fi asemănătoare cu cea pentru Level 0 din Figura 3.1).

```
def nivel_0(x):
    x[0] = 100
    for i in range(1, 270):
        if x[i-1] > 0.32:
            x[i] = round(random.uniform(x[i-1] - 0.32, x[i-1]),
                3)
        else:
            x[i] = round(random.uniform(0, x[i - 1]), 3)
    x[270] = 100
    for i in range(271, 420):
        if 0.4 < x[i - 1] < 20:
            x[i] = round(random.uniform(x[i - 1] - 0.4, x[i-1]),
                3)
        elif x[i - 1] >= 20:
            x[i] = round(random.uniform(x[i - 1] - 2.5, x[i-1]),
                3)
        else:
            x[i] = round(random.uniform(0, x[i - 1]), 3)
    x[420] = 100
    for i in range(421, 570):
        if 0.4 < x[i-1] < 20:
            x[i] = round(random.uniform(x[i - 1] - 0.4, x[i-1]),
                3)
        elif x[i-1] >= 20:
            x[i] = round(random.uniform(x[i-1] - 3, x[i-1]), 3)
        else:
            x[i] = round(random.uniform(0, x[i - 1]), 3)
    return x
```

Variabila x are 570 de elemente. Vom defini funcţia **medii_x(x)** care ne va calcula valoarea medie a intensităţilor radiaţilor pentru fiecare din cele 3 structuri.

```
def medii_x(x):
    media_p = round(sum(x[0:270])/270, 1)
```

```
    media_r = round(sum(x[270:420])/150, 1)
    media_v = round(sum(x[420:570])/150, 1)
    return [media_p, media_r, media_v]
```

Vom avea nevoie şi de funcţia **scalarizare(x, w)** care ne va ajuta sa transformăm o soluţie vector x într-un scalar (scalarizăm mediile fiecărei secţiuni prostată, rect şi vezică, folosind ponderile w).

```
def scalarizare(x, w):
    medii = medii_x(x)
    return medii[0]*w[0] + medii[1]*w[1] + medii[2]*w[2]
```

În continuare, ne vom ocupa pe rând de fiecare nivel ales pentru optimizare. Intervalele pentru fiecare structură, în care iau valori elementele x_i specifice structurii respective, se vor restrânge aproape la fiecare nivel (ideea de optimalitate Pareto - deoarece structurile sunt foarte apropiate, intensitatea unui fascicul influienţează şi zona din apropierea poziţiei pe care acesta era centrat, deci realizarea unui obiectiv pentru o structură reprezintă un anumit compromis pentru celelalte două). Constrângerile sunt adăugate direct în elementele soluţiei x (sunt modificate valorile x_i în funcţie de noile constrângeri), şi folosindu-se varianta ei actualizată se va merge mai departe de la un nivel la altul. Constrângerile sunt restricţii liniare aplicate elementelor x_i, pentru a rezolva nivelele.

Funcţia **nivel_1(x)** calculează intervalul în care trebuie să se afle intensităţile x_i ale prostatei pentru a păstra variaţia de $+/-5\%$ faţă de doza medie. Apoi, în funcţie de poziţia lui x_i faţă de intervalul calculat, fiecare valoare x_i este actualizată, fiind "mutată" în interval sau modificată cu o raţie r calculată în funcţie de medie şi numărul elementelor corespunzătoare prostatei ce erau de la început în intervalul calculat. Vor fi actualizate şi valorile corespunzătoare rectului şi vezicii, în funcţie de modificările din valorile prostatei, iar la final se vor reordona şi se vor corecta unele posibile erori (folosind funcţia seteaza_zero(x)).

```
def nivel_1(x):
    medie_p = medii_x(x)[0]
    prostata = []
    rect = []
    vezica = []
    err = medie_p / 20  # 5% din medie_p
    nr_in_interval = len([x[j] for j in range(0, 270) if medie_p
        - err <= x[j] <= medie_p + err])
    for i in range(0, 270):
        if x[i] < medie_p - err or x[i] > medie_p + err:
            r = medie_p - x[i]
            while abs(r) > err:
                r = round(r / 2, 2)
            p = x[i]
```

```
            while p < medie_p - err or p > medie_p + err:
                p = p + r
            prostata.append(round(p, 3))
        else:
            r = medie_p / nr_in_interval
            p = x[i] + r
            prostata.append(round(p, 3))
    for i in range(270, 420):
        r_r = ((prostata[i - 150] - x[i - 150]) / 5) * (prostata[
            i-150] - x[i]) / (i - 269)
        r_v = ((prostata[i - 150] - x[i - 150]) / 5) * (prostata[
            i-150] - x[i + 150]) / (i - 269)
        rect.append(round(x[i] + r_r, 3))
        vezica.append(round(x[i + 150] + r_v, 3))
    prostata.sort(reverse=True)
    rect.sort(reverse=True)
    vezica.sort(reverse=True)
    x_1 = seteaza_zero(prostata + rect + vezica)
    return x_1
```

Funcţia **nivel_2(x)** urmăreţe creşterea mediei intensităţii radiaţiilor la prostată la 90 Gy (păstrându-se constrângerea anterioară). Astfel, se calculează o raţie de radiaţie care va fi adăugată fiecărui beamlet centrat pe prostată, iar în funcţie de această raţie, a depărtării de centrul tumorii (unde radiaţia e mai puternică) şi a diferenţei de radiaţie dintre fasciculul curent şi cel corespunzător poziţiei respective din celelalte două structuri, se vor actualiza şi celelalte elemente ale planificării x.

```
def nivel_2(x):
    medie_p = medii_x(x)[0]
    r_p = 90 - medie_p  # ratie pentru prostata
    prostata = [round(x[i] + r_p, 3) for i in range(270)]
    rect = [round(x[i] + r_p * (prostata[i-150] - x[i]) / (i -
        269) / 2, 3) for i in range(270, 420)]
    vezica = [round(x[i] + r_p * (prostata[i-300] - x[i]) / (i -
        419) / 2, 3) for i in range(420, 570)]
    prostata.sort(reverse=True)
    rect.sort(reverse=True)
    vezica.sort(reverse=True)
    x_2 = seteaza_zero(prostata + rect + vezica)
    return x_2
```

Funcţia **nivel_3(x)** foloseşte o funcţie (**minimizeaza_rect(x, rect)**) care minimizează radiaţia primită de rect (există o anumită limită până unde pot fi minimizate intensităţile într-o interaţie, ţinându-se cont de depărtarea de centrul tumorii - pentru ca tratamentul să reuşească, tumora trebuie să primească o anumită cantitate de radiaţii). Funcţia nivel_3(x) apelează funcţia minimizeaza_rect(x, rect) de fiecare dată când unul din cele 5 criterii prezentate în tabel la acest nivel nu este satisfăcut (fiecare criteriu e verificat o singură

dată). La final, în funcţie de modificările facute şi de poziţie, se ajustează şi intensităţile fasciculelor centrate asupra rectului şi vezicii.

```
def nivel_3(x):
    rect = x[270:420]
    if len([x[i] for i in range(270, 420) if x[i] > 65]) > int
        (150 * 50 / 100):
        rect = minimizeaza_rect(x, rect)
    if len([rect[i] for i in range(150) if rect[i] > 70]) > int
        (150 * 35 / 100):
        rect = minimizeaza_rect(x, rect)
    if len([rect[i] for i in range(150) if rect[i] > 75]) > int
        (150 * 25 / 100):
        rect = minimizeaza_rect(x, rect)
    if len([rect[i] for i in range(150) if rect[i] > 80]) > int
        (150 * 15 / 100):
        rect = minimizeaza_rect(x, rect)
    if len([rect[i] for i in range(150) if rect[i] > 85.2]) > 0:
        rect = minimizeaza_rect(x, rect)
    prostata = x[:120] + [round(x[i] - (x[i + 150] - rect[i -
        120]) / (i - 119) / 2, 3) for i in range(120, 270)]
    vezica = [round(x[i] - (x[i - 150] - rect[i - 420]) / (i -
        419) / 2, 3) for i in range(420, 570)]
    prostata.sort(reverse=True)
    vezica.sort(reverse=True)
    x_3 = seteaza_zero(prostata + rect + vezica)
    return x_3

def minimizeaza_rect(x, rect):
    rect_update = []
    for i in range(150):
        dif_max = (x[i + 120] - rect[i]) / 3
        if dif_max < 0:
            rect_update.append(round((rect[i] + dif_max * 3), 3))
        else:
            if i > 75:
                indice_modificare = 1 - i / 150
            else:
                indice_modificare = 1 - (150 - i) / 150
            p = dif_max * indice_modificare
            rect_update.append(round((rect[i] - p), 3))
    rect_update.sort(reverse=True)
    return rect_update
```

Funcţia **nivel_4(x)** se ocupă în special pe fasciculele centrate pe radiaţia vezicii şi aplică un algoritm similar cu minimizare_rect(x, rext), minimizând intensităţile. Valoarea nouă se calculează în funcţie de diferenţa dintre intensitatea fasciculului curent (centrat pe vezică) şi o valoare corespunzătoare a prostatei (ne dorim ca iradiera la prostată să fie afectată cât mai puţin prin această minimizare), a depărtării de tumoră (corespunde parametrului

indice_modificare), iar în urmă, actualizează şi intensităţile radiaţiilor celorlalte două structuri.

```
def nivel_4(x):
    vezica = []
    for i in range(420, 570):
        dif_max = (x[i - 300] - x[i]) / 3
        if dif_max < 0:
            vezica.append(round((x[i] + dif_max * 3), 3))
        else:
            indice_modificare = 1 - (i - 420) / 150
            p = dif_max * indice_modificare
            vezica.append(round((x[i] - p), 3))
    prostata = x[:120] + [round(x[i] - (x[i + 300] - vezica[i -
        120]) / (i - 119) / 2, 3) for i in range(120, 270)]
    rect = [round(x[i] - (x[i + 150] - vezica[i - 270]) / (i -
        269) / 2, 3) for i in range(270, 420)]
    prostata.sort(reverse=True)
    rect.sort(reverse=True)
    vezica.sort(reverse=True)
    x_4 = seteaza_zero(prostata + rect + vezica)
    return x_4
```

Funcţia **main()** gestionează aplicarea celor 4 nivele de optimizare pentru fiecare rulare a algoritmilor:

- deoarece soluţia iniţială este generată random după anumite constrângeri, rezultatele obţinute pot să difere de la o iteraţie la alta (putem alege pentru câte rulări să aplicăm algoritmul);
- pentru a păstra constrângerile anterioare, trecerea la un nou nivel al optimizării se face folosind soluţia obţinută în pasul anterior (valorile lui x au fost modificate astfel încât să respecte constrângerile noi adăugate);
- se afişează pe ecran câteva elemente importante pentru fiecare iteraţie;
- se afişează un tabel cu rezultatele centralizate (cele două coloane reprezintă scalarizări ale soluţiilor, obţinute prin ponderi diferite - w si $w2$) se alege valoarea minimă a lui F (minimul se alege în funcţie de scalarizarea obţinută folosind ponderile w, adică F minim, nu F2) din toate soluţiile obţinute în urma iteraţiilor efectuate, se afişează valoarea ei, iar în fişierul "plan.txt" se salvează valorile lui x, ce reprezintă planul optim obţinut.

```
def main():
    try:
        n = int(input("Dati numarul de rulari: "))
        w = [27, 15, 15]
        w2 = [100, 50, 30]
        solutii = []
        for i in range(n):
            x = [0 for _ in range(570)]
```

```
            x_0 = nivel_0(x)
            x_1 = nivel_1(x_0)
            x_2 = nivel_2(x_1)
            x_3 = nivel_3(x_2)
            x_4 = nivel_4(x_3)
            print("Nivelul --- mediile per x ----- prostata
                --------- rect ------------ vezica")
            print(f"Nivelul 1: {medii_x(x_1)}, [{min(x_1[:270])},
                 {max(x_1[:270])}] - [{min(x_1[270:420])}, {max(
                x_1[270:420])}] - [{min(x_1[420:570])}, {max(x_1
                [420:570])}]")
            print(f"Nivelul 2: {medii_x(x_2)}, [{min(x_2[:270])},
                 {max(x_2[:270])}] - [{min(x_2[270:420])}, {max(
                x_2[270:420])}] - [{min(x_2[420:570])}, {max(x_2
                [420:570])}]")
            print(f"Nivelul 3: {medii_x(x_3)}, [{min(x_3[:270])},
                 {max(x_3[:270])}] - [{min(x_3[270:420])}, {max(
                x_3[270:420])}] - [{min(x_3[420:570])}, {max(x_3
                [420:570])}]")
            print(f"Nivelul 4: {medii_x(x_4)}, [{min(x_4[:270])},
                 {max(x_4[:270])}] - [{min(x_4[270:420])}, {max(
                x_4[270:420])}] - [{min(x_4[420:570])}, {max(x_4
                [420:570])}]")
            F = [scalarizare(x_1, w), scalarizare(x_2, w),
                scalarizare(x_3, w), scalarizare(x_4, w)]
            F2 = [scalarizare(x_1, w2), scalarizare(x_2, w2),
                scalarizare(x_3, w2), scalarizare(x_4, w2)]
            print(f" {F}, {F2}")
            solutii.append([F, F2, x_4])
        solutii_finale = ordoneaza_lexicografic_solutiile(solutii
            [:])
        print("REZULTATE:")
        for sol in solutii:
            print(f"{sol[0]}  |  {sol[1]}")
        solutie_finala = solutii_finale[0]
        print(f"F = {solutie_finala[0]}")
        with open('plan.txt', 'w') as file:
            file.write(str(solutie_finala[2]))
    except Exception as e:
        print(e)
```

Funcţia **ordoneaza_lexicografic_solutiile(solutii)** primeşte o listă cu n perechi [F, F2, x] obţinute în urma procesului de optimizare aplicat de n ori, şi ordonează lista lexicografic în funcţie de vectorii F (valorile funcţiilor ce formează vectorul F sunt obţinute scalarizând cu ponderile $w = (27, 15, 15)$, soluţiile obţinute după fiecare nivel de optimizare).

```
def ordoneaza_lexicografic_solutiile(solutii):
    for i in range(len(solutii) - 1):
        for j in range(i + 1, len(solutii)):
            F1 = solutii[i][0]
```

```
            F2 = solutii[j][0]
            F2_mai_mic = False
            for k in range(4):
                if F2[k] < F1[k]:
                    F2_mai_mic = True
                    break
                elif F2[k] > F1[k]:
                    break
            if F2_mai_mic:
                solutii[i], solutii[j] = solutii[j], solutii[i]
    return solutii
```

3.2.4. *Prezentarea şi interpretarea rezultatelor obţinute.*

```
Dati numarul de rulari: 1
Nivelul --- mediile per x ----- prostata --------- rect ------------ vezica
Nivelul 1: [78.3, 35.3, 28.7], [74.142, 83.116] - [4.891, 97.059] - [2.372, 95.114]
Nivelul 2: [90.0, 38.7, 33.2], [85.842, 94.816] - [8.048, 92.024] - [5.627, 91.843]
Nivelul 3: [90.0, 34.2, 33.2], [85.84, 94.816] - [7.445, 91.807] - [5.625, 91.734]
Nivelul 4: [89.9, 34.2, 26.2], [85.839, 94.816] - [7.444, 91.803] - [5.394, 91.725]
 [3074.1, 3508.5, 3441.0, 3333.3], [10456.0, 11931.0, 11706.0, 11486.0]
REZULTATE:
[3074.1, 3508.5, 3441.0, 3333.3]  |  [10456.0, 11931.0, 11706.0, 11486.0]
F = [3074.1, 3508.5, 3441.0, 3333.3]
```

Figura. 3.2 – Rezultate pentru n = 1 iteraţii

În Figura (3.2) se poate observa ce se afişează pe ecran pentru rularea algoritmului nostru o singură dată (⇒ o singură soluţie). În prima secţiune avem prezentate câteva rezultate importante pentru fiecare nivel:

- *Nivelul* → nivelurile reprezentate de fiecare criteriu stabilit, pentru care vom afișa câteva valori importante;
- *mediile per x* → media intensităților radiațiilor primite, în această ordine, de către prostată, rect și vezică urinară, pentru fiecare nivel;
- *prostata* → intervalul în care se află radiațiile x_i pentru prostată;
- *rect* → intervalul în care se află radiațiile x_i pentru rect;
- *vezica* → intervalul în care se află radiațiile x_i pentru vezica urinară;

Observaţia 5. *Se poate vedea cum intervalele din ultimele* 3 *coloane sunt "tăiate" de la un nivel la altul, ţinând cont şi de scopurile fiecărui nivel.*

Imediat după cele 4 niveluri avem soluţiile F şi F2, separate prin virgulă. Acestea sunt calculate scalarizând mediile obţinute la fiecare nivel, pentru F folosind ponderile $w = (27, 15, 15)$, iar pentru F2 folosind ponderile $w2 = (100, 50, 30)$.

În secţiunea *REZULTATE* se centralizează rezultatele obţinute pentru toate cele n rulări (se noteaza pe prima coloană valorile lui F, iar pe a doua coloană

valorile lui F2). La final, F reprezintă valoarea minimă dintre F-urile obţinute în cele n rulări (calculată cu ponderea $w \rightarrow$ vezi şi Figura (3.3)), soluţia x corespunzătoare fiind reţinută în fişierul "plan.txt".

Rulând algoritmul prezentat mai sus pentru două iteraţii, obţinem soluţiile prezentate astfel:

```
Dati numarul de rulari: 2
Nivelul --- mediile per x ----- prostata --------- rect ------------ vezica
Nivelul 1: [77.6, 32.9, 25.4], [73.462, 82.744] - [5.099, 94.909] - [1.927, 95.128]
Nivelul 2: [90.0, 37.1, 31.0], [85.862, 95.144] - [8.437, 91.856] - [5.396, 92.161]
Nivelul 3: [89.9, 32.5, 31.0], [85.86, 95.144] - [7.976, 91.794] - [5.394, 92.13]
Nivelul 4: [89.9, 32.4, 23.6], [85.859, 95.144] - [7.975, 91.748] - [3.463, 91.788]
 [2969.7, 3451.5, 3379.8, 3267.3], [10167.0, 11785.0, 11545.0, 11318.0]
Nivelul --- mediile per x ----- prostata --------- rect ------------ vezica
Nivelul 1: [78.9, 32.9, 28.1], [74.68, 83.976] - [4.335, 97.03] - [1.838, 97.166]
Nivelul 2: [90.0, 36.4, 32.2], [85.78, 95.076] - [7.348, 92.279] - [4.944, 91.827]
Nivelul 3: [89.9, 31.7, 32.2], [85.779, 95.076] - [7.079, 91.728] - [4.943, 91.552]
Nivelul 4: [89.9, 31.6, 25.1], [85.778, 95.076] - [7.078, 91.71] - [3.846, 91.499]
 [3045.3, 3459.0, 3385.8, 3277.8], [10378.0, 11786.0, 11541.0, 11323.0]
REZULTATE:
[2969.7, 3451.5, 3379.8, 3267.3]  |  [10167.0, 11785.0, 11545.0, 11318.0]
[3045.3, 3459.0, 3385.8, 3277.8]  |  [10378.0, 11786.0, 11541.0, 11323.0]
F = [2969.7, 3451.5, 3379.8, 3267.3]
```

Figura. 3.3 – Rezultate pentru n = 2 iteraţii

Observăm că F reprezintă valoarea minimă, optimă lexicografic a funcţiei scop, din cele două iteraţii efectuate.

În continuare vom rula algoritmul proiectat pentru ponderi diferite şi vom face câteva observaţii asupra rezultatelor obţinute.

	F ($w = (27, 15, 15)$)	**F2** ($w2 = (100, 50, 30)$)
S_1	[2989.5, 3435.0, 3360.3, 3250.8]	[10202.0, 11708.0, 11458.0, 11237.0]
S_2	[3099.3, 3492.0, 3345.3, 3237.3]	[10571.0, 11902.0, 11414.0, 11196.0]
S_3	[3020.4, 3432.0, 3360.0, 3247.8]	[10316.0, 11710.0, 11470.0, 11239.0]
S_4	[3023.1, 3469.5, 3397.5, 3289.5]	[10312.0, 11821.0, 11583.0, 11365.0]
S_5	[3037.5, 3486.0, 3414.0, 3312.0]	[10319.0, 11844.0, 11606.0, 11400.0]
S_6	[3154.8, 3522.0, 3451.5, 3355.5]	[10647.0, 11910.0, 11675.0, 11481.0]

Tabela 2 – Scalarizări cu ponderi diferite:$w = (27, 15, 15)$ vs $w2 = (100, 50, 30)$

Ordonând în funcţie de F, obţinem următoarea ordine pentru soluţii:

$$S_1 \leq S_3 \leq S_4 \leq S_5 \leq S_2 \leq S_6.$$

Ordonând în funcţie de F2, obţinem următoarea ordine pentru soluţii:

$$S_1 \leq S_4 \leq S_3 \leq S_5 \leq S_2 \leq S_6.$$

Ordonările soluţiilor corespunzătoare celor doi vectori de ponderi $w = (27, 15, 15)$ şi $w2 = (100, 50, 30)$ sunt uşor diferite (soluţiile S_3 şi S_4 sunt interschimbate), dar soluţia optimă lexicografic, în ambele cazuri, este S_1.

	F ($w = (27, 15, 15)$)	**F2** ($w2 = (50, 40, 30)$)
S_1	[3013.5, 3412.5, 3336.3, 3226.8]	[6008.0, 6802.0, 6601.0, 6381.0]
S_2	[2995.5, 3459.0, 3390.0, 3273.3]	[6019.0, 6937.0, 6753.0, 6519.0]
S_3	[3089.4, 3493.5, 3421.5, 3322.5]	[6183.0, 6984.0, 6792.0, 6594.0]
S_4	[3044.1, 3489.0, 3415.5, 3315.0]	[6088.0, 6973.0, 6778.0, 6577.0]
S_5	[3027.0, 3442.5, 3373.5, 3261.0]	[6076.0, 6898.0, 6714.0, 6488.0]
S_6	[3049.5, 3502.5, 3433.5, 3327.0]	[6138.0, 7033.0, 6850.0, 6636.0]

Tabela 3 – Scalarizări cu ponderi diferite:$w = (27, 15, 15)$ vs $w2 = (50, 40, 30)$

Ordonând în funcţie de F, obţinem următoarea ordine pentru soluţii:

$$S_2 \leq S_1 \leq S_5 \leq S_4 \leq S_6 \leq S_3.$$

Ordonând în funcţie de F2, obţinem următoarea ordine pentru soluţii:

$$S_1 \leq S_2 \leq S_5 \leq S_4 \leq S_6 \leq S_3.$$

Ordonările soluţiilor corespunzătoare celor doi vectori de ponderi $w = (27, 15, 15)$ şi $w2 = (50, 40, 30)$, sunt mult mai diferite între ele decât diferenţele care au fost între ordonările realizate cu datele din Tabela (2), iar soluţia optimă lexicografic pentru F este S_2, în timp ce pentru F2 este S_1.

4. CONCLUZII

Optimizarea lexicografică în planificarea tratamentului cu IMRT reflectă procesul intuitiv de luare a deciziilor al medicilor, facilitând definirea clară a problemei de optimizare şi obţinerea unui plan acceptabil din punct de vedere clinic. Această metodă permite medicilor să prioritizeze obiectivele tratamentului, astfel încât obiectivele cele mai importante să fie satisfăcute complet, acceptându-se compromsuri doar la niveluri mai puţin critice, transformând astfel, procesul decizional bazat pe priotităţi, într-o optimizare ordonată.

Cu toate că ar fi nevoie de unele cercetări suplimentare, metoda optimizării lexicografice pare promiţătoare pentru îmbunătăţirea planificării tratamentelor clinice IMRT, oferind o structură clară şi uşor de utilizat pentru deciziile progresive.

BIBLIOGRAFIE

[1] A. Taylor, M.E.B. Powell, *Intensity-modulated radiotherapy—what is it?*, online, Martie 2004.

[2] Ben W. Fischer-Valuck, Yuan James Rao, Jeff M. Michalski, *Intensity-modulated radiotherapy for prostate cancer*, Department of Radiation Oncology, Washington University School of Medicine, St. Louis, USA, Iulie 2018.

[3] Byungchul Cho, PhD, *Intensity-modulated radiation therapy: a review with a physics perspective*, Department of Radiation Oncology, Asan Medical Center, University of Ulsan College of Medicine, Seoul, Korea, Martie 2018.

[4]

[5] Lars-Ake Lindahl, *Convexity and Optimization*, Uppsala, April 2016.

[6] M.M. Makela, Y. Nikulin, *On cone Characterisation of Strong and Lexicographic Optimality in Convex Multiobjective Optimization*, Springer Science + Business Media, 2009.

[7] *Metode de Optimizare - V.4. Condiţia de regularitate Slater*, `https://www.utgjiu.ro/math/mbuneci/book/mo2007/c09.pdf`.

[8] *Multi-Objective Optimization*, `https://engineering.purdue.edu/~sudhoff/ee630/Lecture09.pdf`.

[9] Natalya Semenova, Maria Lomaha, *Vector Convex Optimization Problems: Lexicographic Optimality and Solvability, Method of Solving*, International Scientific Symposium ≪Intelligent Solutions≫, Kyiv-Uzhhorod, Ukraine, Septembrie 2021.

[10] Nicolae Popovici, *Tehnici de optimizare*, UBB, Cluj-Napoca, 2020.

[11] *Optimization Techniques - Lecture 06 - Polyhedral Sets*, Industrial Engineering and Operations Research, Indian Institute of Technology Bombay, Aprilie 2019.

[12] *Pareto Optimality*, `https://web.stanford.edu/group/sisl/k12/optimization/MO-unit5-pdfs/5.8Pareto.pdf`.

[13] Trif Tiberiu, *Analiză matematică 2 - Calcul diferenţial şi integral în* $\mathbb{R}^n$, UBB, Cluj-Napoca.

[14] `https://ro.wikipedia.org/wiki/Rela%C8%9Bie_de_ordine`.

[15] `https://en.wikipedia.org/wiki/Multi-objective_optimization#A_priori_methods`.

[16] `https://medium.com/@personxy/multi-objective-optimization-via-scalarization-approach-1e1e054506b6`.

[17] `https://ocw.mit.edu/courses/6-253-convex-analysis-and-optimization-spring-2012/69136e797daa75d9481c21d1f6b84a31_MIT6_253S12_lec05.pdf`.

Faculty of Mathematics and Computer Science
"Babeş-Bolyai" University
Str. Kogălniceanu, no. 1
400084 Cluj-Napoca, Romania