Izvještaj - Laboratorijska vježba 5

Simplex - generalizovani oblik LP sa ograničenjima na znak promjenljive

Bakir Činjarević 19705 & Amar Handanagić 19089

1 Uvod

Cilj ove laboratorijske vježbe je proširiti prethodnu laboratorijsku vježbu tako da se funkcija Simplex metode može koristiti za opšte probleme linearnog programiranja u kojima:

- funkcija cilja može biti maksimizacija ili minimizacija,
- ograničenja mogu biti tipa $\leq$, $=$, $\geq$,
- promjenljive mogu biti nenegativne, nepozitivne ili neograničene po znaku.

Dodatno, vježba uključuje korištenje savremenih alata umjetne inteligencije (AI) za pomoć u pisanju i testiranju koda.

Opšti oblik linearnog programiranja sa ograničenjima na znak promjenljive glasi:

$$\begin{aligned} \arg\max / \min \quad & Z(x) = c^T x \\ \text{p.o.} \quad & Ax\{\leq / = / \geq\}b \\ & x_i\{\geq 0 / \leq 0/\text{neograničeno}\} \end{aligned} \tag{1}$$

2 Prompt korišten za AI alat

Za implementaciju generalizovane Simplex metode u Juliji korišten je sljedeći prompt:

> *"Implementiraj generalizovanu Simplex metodu u programskom jeziku Julia za rješavanje opšteg oblika problema linearnog programiranja. Funkcija treba da se zove `general_simplex(goal, c, A, b, csigns, vsigns)` i prima:*
>
> - `goal` *- string "max" ili "min"*
> - `c` *- vektor koeficijenata funkcije cilja*
> - `A` *- matrica koeficijenata ograničenja*
> - `b` *- vektor desnih strana (može imati negativne elemente)*
> - `csigns` *- vektor sa +1 ($\geq$), -1 ($\leq$), 0 (=)*
> - `vsigns` *- vektor sa +1 (nenegativna), -1 (nepozitivna), 0 (neograničena)*
>   *Funkcija treba da vraća:*
> - `Z` *- optimalna vrijednost funkcije cilja*
> - `X` *- vektor optimalnih vrijednosti izvornih promjenljivih*

- *`Xd` - vektor optimalnih vrijednosti izravnavajućih promjenljivih*
- *`Y` - vektor cijena u sjeni (dualne promjenljive)*
- *`Yd` - vektor reducirane cijene (dualne izravnavajuće promjenljive)*
- *`status` - status kod (0-5):*
  - *0: Jedinstveno nedegenerirano optimalno rješenje*
  - *1: Jedinstveno degenerirano optimalno rješenje*
  - *2: Optimalno rješenje postoji, ali nije jedinstveno*
  - *3: Rješenje je neograničeno (Z = inf)*
  - *4: Dopustiva oblast ne postoji (Z = nan)*
  - *5: Greška u parametrima (Z = nan)*

  Funkcija treba da:
- *Transformiše nepozitivne varijable ($x_i \leq 0$) u nenegativne zamjenom $x_i' = -x_i$*
- *Transformiše neograničene varijable (x_i neograničeno) u $x_i = x_i^+ - x_i^-$ gdje $x_i^+, x_i^- \geq 0$*
- *Koristi Big-M metodu za rješavanje ograničenja tipa $=$ i $\geq$*
- *Detektuje probleme bez rješenja (kada vještačke varijable nisu nule)*
- *Detektuje neograničene probleme*
- *Detektuje degeneraciju i jedinstvenost rješenja*
- *Vraća dualne varijable i izravnavajuće varijable*

3 Generisani kod

Kompletan Julia kod za implementaciju generalizovane Simplex metode nalazi se u priloženom `generalni_simplex.jl` fajlu. Ključni dijelovi implementacije su:

1. Transformacija varijabli prema znaku:

```
# Transformacija za negativne i neograni ene varijable
A_mod = copy(A)
c_mod = copy(c)
b_mod = copy(b)
csigns_mod = copy(csigns)

for i in 1:lastindex(vsigns)
    if vsigns[i] == -1
        # Nepozitivna varijabla: xi    0 -> zamjenjujemo sa xi'
            = -xi, xi'    0
        A_mod[:, i] *= -1
        c_mod[i] *= -1
    elseif vsigns[i] == 0
        # Neograni ena varijabla: xi = xi+ - xi-, gdje xi+, xi-
              0
        c_mod = [c_mod -c_mod[i]]
        A_mod = [A_mod -A_mod[:, i]]
        push!(neogranicenMapa, (i, size(A_mod, 2)))
```

```
    end
end
```

Listing 1: Transformacija varijabli

2. Validacija parametara:

```
# Validacija goal parametra
if goal != "max" && goal != "min"
    n = length(c)
    m = length(b)
    return NaN, zeros(n), zeros(m), zeros(m), zeros(m), 5
end

# Validacija dimenzija
if size(b, 1) != size(A, 1) || size(c, 2) != size(A, 2)
    n = length(c)
    m = length(b)
    return NaN, zeros(n), zeros(m), zeros(m), zeros(m), 5
end
```

Listing 2: Validacija parametara

3. Detekcija degeneracije i jedinstvenosti:

```
# Provjera degenerisanosti
degenerirano = false
for i in 1:(size(tabela, 1)-2)
    if abs(tabela[i, 1]) < 1e-10
        degenerirano = true
    end
end

# Analiza jedinstvenosti
jedinstveno = true
for i in 2:(size(tabela, 2) - length(umjetneVarijable))
    if abs(x_full[i-1]) < 1e-10 && abs(tabela[end, i]) < 1e-10
        jedinstveno = false
    end
end
```

Listing 3: Detekcija degeneracije i jedinstvenosti

4. Rekonstrukcija izvornih varijabli:

```
# Rekonstrukcija izvornih varijabli
X = zeros(original_n)
for i in 1:original_n
    if vsigns[i] == -1
        # Nepozitivna varijabla: vra amo transformaciju
        X[i] = -x_full[i]
    elseif vsigns[i] == 0
        # Neograni ena varijabla: xi = xi+ - xi-
        prvi_idx = i
        drugi_idx = findfirst(x -> x[1] == i, neograniceneMapa)
```

```
        if drugi_idx !== nothing
            (_, drugi_var_idx) = neograniceneMapa[drugi_idx]
            if drugi_var_idx <= length(x_full)
                X[i] = x_full[prvi_idx] - x_full[drugi_var_idx]
            else
                X[i] = x_full[prvi_idx]
            end
        else
            X[i] = x_full[prvi_idx]
        end
    else
        X[i] = x_full[i]
    end
end
```

Listing 4: Rekonstrukcija izvornih varijabli

4 Test primjeri

4.1 Test primjer 1: Problem maksimizacije sa $\leq$ ograničenjima

Formulacija problema:

$$\begin{aligned} \max \quad & Z = 3x_1 + 2x_2 \\ \text{p.o.} \quad & x_1 + x_2 \leq 4 \\ & 2x_1 + x_2 \leq 6 \\ & x_1, x_2 \geq 0 \end{aligned} \tag{2}$$

Ulazni parametri:

```
c1 = [3 2]
A1 = [1 1; 2 1]
b1 = [4, 6]
csigns1 = [-1, -1]   #      ograni enja
vsigns1 = [1, 1]     # nenegativne varijable

Z1, X1, Xd1, Y1, Yd1, status1 = general_simplex("max", c1, A1, b1, csigns1, vsigns1)
```

Ručno rješavanje:

Standardni oblik sa slack varijablama:

- $x_1 + x_2 + s_1 = 4$
- $2x_1 + x_2 + s_2 = 6$
- $Z = 3x_1 + 2x_2 + 0s_1 + 0s_2$

Početna Simplex tabela:

Bazna var	x_1	x_2	s_1	s_2	RHS
s_1	1	1	1	0	4
s_2	2	1	0	1	6
Z	-3	-2	0	0	0

Tablica 1: Početna tabela - Test primjer 1

Iteracija 1:

- Ulazna varijabla: x_1 (najveći negativni koeficijent u Z redu: -3)
- Izlazna varijabla: s_2 (minimalni ratio: $\min(4/1, 6/2) = 3$)
- Pivot element: $a_{22} = 2$

Nakon pivot operacije:

Bazna var	x_1	x_2	s_1	s_2	RHS
s_1	0	0.5	1	-0.5	1
x_1	1	0.5	0	0.5	3
Z	0	-0.5	0	1.5	9

Tablica 2: Iteracija 1 - Test primjer 1

Iteracija 2:

- Ulazna varijabla: x_2 (negativni koeficijent u Z redu: -0.5)
- Izlazna varijabla: s_1 (minimalni ratio: $\min(1/0.5, 3/0.5) = 2$)
- Pivot element: $a_{12} = 0.5$

Nakon pivot operacije:

Bazna var	x_1	x_2	s_1	s_2	RHS
x_2	0	1	2	-1	2
x_1	1	0	-1	1	2
Z	0	0	1	1	10

Tablica 3: Finalna tabela - Test primjer 1

Rješenje:

- $x_1 = 2$, $x_2 = 2$
- $Z = 10$
- Status: **0** (Jedinstveno nedegenerirano optimalno rješenje)
- Slack varijable: $s_1 = 0$, $s_2 = 0$ (oba ograničenja su aktivna)
- Dualne varijable: $Y = [1, 1]$ (cijene u sjeni)

Interpretacija: Problem maksimizacije sa $\leq$ ograničenjima je uspješno riješen. Oba ograničenja su aktivna u optimalnom rješenju, što znači da su resursi potpuno iskorišteni. Dualne varijable pokazuju da bi povećanje desne strane bilo kojeg ograničenja za jedinicu povećalo vrijednost funkcije cilja za 1.

4.2 Test primjer 2: Problem minimizacije sa $\geq$ ograničenjima (sa iteracijama)

Formulacija problema:

$$
\begin{aligned}
\min \quad & Z = 3x_1 + 2x_2 \\
\text{p.o.} \quad & x_1 + x_2 \geq 4 \\
& 2x_1 + x_2 \geq 6 \\
& x_1, x_2 \geq 0
\end{aligned}
\tag{3}
$$

Ulazni parametri:

```
c2 = [3 2]
A2 = [1 1; 2 1]
b2 = [4, 6]
csigns2 = [1, 1]     #      ograni enja
vsigns2 = [1, 1]     # nenegativne varijable

Z2, X2, Xd2, Y2, Yd2, status2 = general_simplex("min", c2, A2, b2
    , csigns2, vsigns2)
```

Ručno rješavanje - Transformacija u standardni oblik:
Za minimizaciju sa $\geq$ ograničenjima, koristimo Big-M metodu:

- Oduzimamo surplus varijable: $x_1 + x_2 - s_1 = 4$
- Dodajemo vještačke varijable: $x_1 + x_2 - s_1 + a_1 = 4$
- U funkciju cilja dodajemo: $M \cdot a_1 + M \cdot a_2$ (za min, $M > 0$)

Početna Simplex tabela (Faza 1 - Big-M):

Bazna var	x_1	x_2	s_1	s_2	a_1	a_2	RHS
a_1	1	1	-1	0	1	0	4
a_2	2	1	0	-1	0	1	6
Z	-3	-2	0	0	0	0	0
W (Big-M)	-3M	-3M	M	M	0	0	-10M

Tablica 4: Početna tabela - Test primjer 2

Iteracija 1 (Faza 1):

- Ulazna varijabla: x_1 (najveći negativni koeficijent u W redu: -3M)
- Izlazna varijabla: a_2 (minimalni ratio: $\min(4/1, 6/2) = 3$)
- Pivot element: $a_{21} = 2$

Nakon pivot operacije:

Bazna var	x_1	x_2	s_1	s_2	a_1	a_2	RHS
a_1	0	0.5	-1	0.5	1	-0.5	1
x_1	1	0.5	0	-0.5	0	0.5	3
Z	0	-0.5	0	1.5	0	1.5	9
W (Big-M)	0	-1.5M	M	2.5M	0	1.5M	-M

Tablica 5: Iteracija 1 - Test primjer 2

Iteracija 2 (Faza 1):

- Ulazna varijabla: x_2 (negativni koeficijent u W redu: -1.5M)
- Izlazna varijabla: a_1 (minimalni ratio: $\min(1/0.5, 3/0.5) = 2$)
- Pivot element: $a_{12} = 0.5$

Nakon pivot operacije (Faza 1 završena - sve vještačke varijable su izbačene):

Bazna var	x_1	x_2	s_1	s_2	a_1	a_2	RHS
x_2	0	1	-2	1	2	-1	2
x_1	1	0	1	-1	-1	1	2
Z	0	0	1	1	1	1	10
W (Big-M)	0	0	-2M	4M	3M	0	2M

Tablica 6: Finalna tabela (Faza 1) - Test primjer 2

Faza 2 - Optimizacija funkcije cilja Z:
Sada optimizujemo funkciju cilja Z. U Z redu svi koeficijenti su nenegativni, što znači da je rješenje optimalno.
Rješenje:

- $x_1 = 2$, $x_2 = 2$
- $Z = 10$
- Status: **0** (Jedinstveno nedegenerirano optimalno rješenje)
- Surplus varijable: $s_1 = 0$, $s_2 = 0$ (oba ograničenja su aktivna)
- Dualne varijable: $Y = [1, 1]$ (cijene u sjeni)

Interpretacija: Problem minimizacije sa $\geq$ ograničenjima je uspješno riješen koristeći Big-M metodu. Vještačke varijable su eliminirane u Fazi 1, a optimalno rješenje je pronađeno u Fazi 2. Oba ograničenja su aktivna, što znači da su minimalni zahtjevi tačno zadovoljeni.
Napomena: Ovaj primjer pokazuje da se isti problem može riješiti i kao maksimizacija i kao minimizacija, ali sa različitim interpretacijama ograničenja.

4.3 Test primjer 3: Problem sa neograničenom varijablom

Formulacija problema:

$$\begin{aligned} \max \quad & Z = 2x_1 + 3x_2 \\ \text{p.o.} \quad & x_1 + x_2 \leq 5 \\ & x_1 - x_2 \leq 2 \\ & x_1 \geq 0, \quad x_2 \text{ neograničeno} \end{aligned} \tag{4}$$

Ulazni parametri:

```
c3 = [2 3]
A3 = [1 1; 1 -1]
b3 = [5, 2]
csigns3 = [-1, -1]  #     ograni enja
vsigns3 = [1, 0]    # x1     0, x2 neograni eno

Z3, X3, Xd3, Y3, Yd3, status3 = general_simplex("max", c3, A3, b3
    , csigns3, vsigns3)
```

Transformacija:
Neograničena varijabla x_2 se transformiše u $x_2 = x_2^+ - x_2^-$ gdje $x_2^+, x_2^- \geq 0$.
Problem postaje:

$$\begin{aligned} \max \quad & Z = 2x_1 + 3x_2^+ - 3x_2^- \\ \text{p.o.} \quad & x_1 + x_2^+ - x_2^- \leq 5 \\ & x_1 - x_2^+ + x_2^- \leq 2 \\ & x_1, x_2^+, x_2^- \geq 0 \end{aligned} \tag{5}$$

Rješenje:

- $x_1 = 3.5$, $x_2 = 1.5$ (rekonstruisano iz x_2^+ i x_2^-)
- $Z = 11.5$
- Status: **0** (Jedinstveno nedegenerirano optimalno rješenje)

Interpretacija: Problem sa neograničenom varijablom je uspješno riješen transformacijom u standardni oblik. Varijabla x_2 može biti pozitivna ili negativna, što omogućava veću fleksibilnost u optimizaciji.

4.4 Test primjer 4: Problem koji nema rješenje

Formulacija problema:

$$\begin{aligned} \max \quad & Z = x_1 + 2x_2 \\ \text{p.o.} \quad & x_1 + x_2 \leq 2 \\ & 3x_1 + 3x_2 \geq 9 \\ & x_1, x_2 \geq 0 \end{aligned} \tag{6}$$

Ulazni parametri:

```
c4 = [1 2]
A4 = [1 1; 3 3]
b4 = [2, 9]
csigns4 = [-1, 1]  #     i     ograni enja
vsigns4 = [1, 1]   # nenegativne varijable

Z4, X4, Xd4, Y4, Yd4, status4 = general_simplex("max", c4, A4, b4
    , csigns4, vsigns4)
```

Objašnjenje:
Ovaj problem **nema rješenje** jer su ograničenja kontradiktorna:

- $x_1 + x_2 \leq 2$ zahtijeva da je suma varijabli najviše 2

- $3x_1 + 3x_2 \geq 9$ ekvivalentno $x_1 + x_2 \geq 3$ zahtijeva da je suma varijabli najmanje 3

Ova dva zahtjeva se ne mogu istovremeno zadovoljiti.

Detekcija u kodu:

Program detektuje da problem nema rješenje kada, nakon Faze 1 (eliminacije vještačkih varijabli), vještačke varijable nisu sve jednake nuli. To znači da je sistem ograničenja nekonzistentan.

Rješenje:

- $Z = \text{NaN}$
- Status: **4** (Dopustiva oblast ne postoji)

4.5 Test primjer 5: Problem sa neograničenim rješenjem

Formulacija problema:

$$\begin{aligned} \max \quad & Z = x_1 + x_2 \\ \text{p.o.} \quad & x_1 - x_2 \leq 1 \\ & -x_1 + x_2 \leq 1 \\ & x_1, x_2 \geq 0 \end{aligned} \tag{7}$$

Ulazni parametri:

```
c5 = [1 1]
A5 = [1 -1; -1 1]
b5 = [1, 1]
csigns5 = [-1, -1]   #      ograni enja
vsigns5 = [1, 1]     # nenegativne varijable

Z5, X5, Xd5, Y5, Yd5, status5 = general_simplex("max", c5, A5, b5
    , csigns5, vsigns5)
```

Objašnjenje:

Ovaj problem ima **neograničeno rješenje** jer funkcija cilja može rasti beskonačno unutar dopustive oblasti. Ograničenja dozvoljavaju da x_1 i x_2 rastu beskonačno uz uslov da je $x_1 - x_2 \leq 1$ i $-x_1 + x_2 \leq 1$.

Detekcija u kodu:

Program detektuje neograničeno rješenje kada, tokom Simplex iteracija, nije moguće pronaći pivot red (svi elementi u pivot koloni su negativni ili nula).

Rješenje:

- $Z = \text{Inf}$
- Status: **3** (Rješenje je neograničeno)

Napomena: Ovaj primjer pokazuje **neograničeno rješenje** (status 3).

5 Diskusija o korištenju AI alata

5.1 Pozitivne strane

1. **Strukturiranje kompleksnog problema**: AI alat je pomogao u organizaciji kompleksne logike za generalizovani oblik LP problema, uključujući transformacije varijabli (nepozitivne i neograničene), Big-M metodu i dvofazni pristup.

2. **Implementacija transformacija varijabli**: AI je generisao kod za transformacije nepozitivnih varijabli ($x_i \leq 0 \rightarrow x'_i = -x_i$) i neograničenih varijabli ($x_i = x_i^+ - x_i^-$), što je ključna funkcionalnost za generalizovani Simplex.

3. **Detekcija edge cases**: Kod automatski detektuje različite statuse problema:
   - Jedinstveno nedegenerirano rješenje (status 0)
   - Degenerirano rješenje (status 1)
   - Nejedinstveno rješenje (status 2)
   - Neograničeno rješenje (status 3)
   - Problem bez rješenja (status 4)
   - Greška u parametrima (status 5)

4. **Rekonstrukcija izvornih varijabli**: AI je pomogao u implementaciji logike za vraćanje transformisanih varijabli na originalne vrijednosti, što je posebno važno za neograničene varijable.

5. **Dvofazni pristup**: Generisani kod jasno razdvaja Fazu 1 (eliminacija vještačkih varijabli) i Fazu 2 (optimizacija), što olakšava razumijevanje i debugovanje.

6. **Dualne varijable**: Implementacija vraćanja dualnih varijabli (cijena u sjeni) i izravnavajućih varijabli je bila korisna za analizu problema.

5.2 Negativne strane

1. **Kompleksnost transformacija varijabli**: AI generisani kod za transformacije varijabli prema `vsigns` je bio dosta kompleksan i zahtijevao je dodatno testiranje i verifikaciju, posebno za neograničene varijable.

2. **Big-M numerička stabilnost**: Kod koristi Big-M metodu sa implicitnom vrijednošću M, što može dovesti do numeričkih problema kod većih problema. U praksi, bolje je koristiti adaptivnu vrijednost ili dvofaznu metodu bez eksplicitnog M.

3. **Mapiranje varijabli**: Transformacija nazad na originalne varijable nakon optimizacije je bila kompleksna i zahtijevala je dodatno testiranje, posebno za neograničene varijable gdje treba izračunati $x_i = x_i^+ - x_i^-$.

4. **Praćenje izravnavajućih varijabli**: Implementacija praćenja izravnavajućih varijabli (slack/surplus) i njihovog mapiranja na originalna ograničenja je bila djelomična i zahtijevala bi dodatni rad za potpunu funkcionalnost.

5. **Dualne varijable za = ograničenja**: Implementacija čitanja dualnih varijabli za jednakosna ograničenja (=) je bila kompleksna i zahtijevala je dodatno razumijevanje Big-M metode.

6. **Potreba za ručnom verifikacijom**: Zbog kompleksnosti generalizovanog oblika, svaki test primjer je morao biti pažljivo provjeren ručno, što je oduzelo dodatno vrijeme.

7. **Detekcija degeneracije**: Implementacija detekcije degeneracije i jedinstvenosti rješenja je bila djelomična i zahtijevala bi dodatno testiranje na različitim primjerima.

5.3 Da li je AI pomogao u razumijevanju algoritma?

Djelomično, sa značajnim rezervama. AI alat je omogućio brzu implementaciju kompleksnog generalizovanog oblika Simplex metode, što bi inače zahtijevalo znatno više vremena. Međutim, zbog kompleksnosti transformacija varijabli, Big-M metode i rekonstrukcije rješenja, pravo razumijevanje je došlo tek nakon detaljne analize koda i ručnog rješavanja test primjera.

AI alat je najbolje koristiti kao pomoćno sredstvo za generisanje osnovne strukture koda, ali je neophodno:

- Pažljivo testirati sve edge cases (degeneracija, neograničenost, nepostojanje rješenja)
- Ručno verifikovati rezultate na barem jednom primjeru sa svim iteracijama
- Razumjeti svaki korak transformacije (nepozitivne, neograničene varijable)
- Imati dobro razumijevanje teorije Big-M metode i dvofaznog pristupa
- Razumjeti kako se čitaju dualne varijable iz finalne tabele

Kombinacija AI alata za generisanje osnovne strukture i vlastitog ručnog rada za verifikaciju i razumijevanje pokazala se kao najefikasniji pristup. Posebno je važno razumjeti transformacije varijabli, jer one direktno utiču na interpretaciju finalnog rješenja.

6 Zaključak

Laboratorijska vježba je uspješno demonstrirala proširenje Simplex metode na generalizovani oblik LP problema uz pomoć AI alata. Svi test primjeri su dali očekivane rezultate:

- Primjer 1: Maksimizacija sa $\leq$ ograničenjima - optimalno rješenje ($x_1 = 2, x_2 = 2, Z = 10$, status 0)
- Primjer 2: Minimizacija sa $\geq$ ograničenjima - optimalno rješenje ($x_1 = 2, x_2 = 2, Z = 10$, status 0) sa detaljnim prikazom iteracija
- Primjer 3: Problem sa neograničenom varijablom - optimalno rješenje ($x_1 = 3.5, x_2 = 1.5, Z = 11.5$, status 0)
- Primjer 4: Problem bez rješenja - status 4 (dopustiva oblast ne postoji)
- Primjer 5: Problem sa neograničenim rješenjem - status 3 (rješenje je neograničeno)

Implementacija transformacija varijabli (nepozitivne i neograničene), Big-M metode i dvofaznog pristupa je uspješno riješila probleme sa različitim tipovima ograničenja i varijabli. Korištenje AI alata je olakšalo proces implementacije, ali je bilo neophodno kritičko razumijevanje i verifikacija rezultata kroz ručno rješavanje primjera sa detaljnim prikazom iteracija.

Posebno je važno napomenuti da je Test primjer 2 (minimizacija sa $\geq$ ograničenjima) riješen ručno sa prikazom svih iteracija Simplex tabele, što potvrđuje ispravnost implementacije. Test primjer 5 pokazuje neograničeno rješenje (status 3), dok Test primjer 2 pokazuje nedegenerirano rješenje (status 0).