UNIVERZITET U NIŠU
ELEKTRONSKI FAKULTET

Metode optimizacije

Seminarski rad

Optimizacije sa ograničenjima

Mentor:

Prof. dr Slađana Marinković

Student:

Svetlana Mančić 1423

Sadržaj

1. Uvod

Rešavanje optimizacionog problema iz prakse zahteva najpre definisanje matematičkog modela. Matematički model podrazumeva identifikaciju ciljne funkcije, čiji se optimum traži, nezavisno promenljivih od kojih vrednost ciljne funkcije zavisi, kao i ograničenja, koja nezavisno promenljive moraju da zadovolje. Nad tako definisanim matematičkim modelom mogu se primeniti različite metode optimizacije.

Kada govorimo o optimizaciji u zavisnosti od oblika ciljne funkcije i ograničenja može se izvršiti podela na: jednokriterijumske i višekriterijumske, optimizacije sa ili bez ograničenja, itd.

U nastavku seminarskog rada biće predstavljena teorijska osnova optimizacije sa ograničenjima, kao i metod kaznenih funkcija, posebno spoljašnjih kaznenih funkcija, koji će se koristiti pri rešavanju zadatka.

Na kraju seminarskog rada biće priloženo rešenje zadatka implementirano u *python* programskom jeziku, uz korišćenje biblioteka *sympy* i *matplotlib*.

2. Optimizacija sa ograničenjima

Ograničenja definišu dopustiv skup problema, zato ukoliko postoje moraju biti uzeta u obzir. Kada se rešavaju problemi sa ograničenjima postoje dva pristupa koja se mogu primeniti. Prvi podrazumeva određivanje optimalnog rešenja uz zadovoljavanje ograničenja, dok drugi podrazumeva transformaciju problema tako da se ograničenja ugrađuju u ciljnu funkciju, te se takav problem rešava kao zadatak bez ograničenja.

Posmatra se problem minimizacije sa ograničenjima, gde su sva ograničenja zadata jednakostima:

$$\min f(x)$$

$$h_k(x) = 0,\ k = 1,...,K$$

$$x = [x_1, \ldots, x_N]^T$$

Gde su funkcije f: $R^n \rightarrow R$ *i* h_k: $R^n \rightarrow R$, $k = 1,\ldots,K$ definisane i diferencijabilne na čitavom prostoru R^n.

Ovakav problem može se rešiti metodom eliminacije promenljivih ili metodom Lagranžovih multiplikatora.

2.1. Lagranžovi multiplikatori

Metoda Lagranžovih multiplikatora daje skup potrebnih uslova za identifikaciju kandidata optimalne tačke optimizacionog problema sa ograničenjima. Ovo se vrši prevođenjem problema sa ograničenjima u ekvivalentni problem bez ograničenja uvođenjem odgovarajućih parametara, koje nazivamo Lagranžovi multiplikatori.

Ako je zadat problem optimizacije funkcije sa N nezavisno promenljivih sa jednim ograničenjem tipa jednakosti:

$$\min f(x)$$

$$h_1(x) = 0$$

$$x = [x_1, \ldots, x_N]^T$$

Metoda Lagranžovih multiplikatora zadati problem prevodi u problem bez ograničenja:

$$\min F(x, \lambda) = f(x) - \lambda h_1(x)$$

gde $F(x, \lambda)$ predstavlja Lagranžovu funkciju, a λ je Lagranžov multiplikator.

Lagranžov metod se može generalizovati na slučaj optimizacije sa više ograničenja tipa jednakosti:

$$\min f(x)$$

$$h_k(x) = 0, k = 1,...,K$$

$$x = [x_1, \ldots, x_N]^T$$

Proširena Lagranžova funkcija u ovom slučaju je:

$$\min F(x, \lambda) = f(x) - \sum_{k=1}^{K} \lambda_k h_k(x)$$

Veličine λ_1, λ_2, ... , λ_k su Lagranžovi multiplikatori, koje je potrebno odrediti zajedno sa promenljivim x_1, x_2, ... , x_N. Izjednačavanjem parcijalnih izvoda funkcije F po x se dobija sistem od N jednačina sa N nepoznatih. Ovom sistemu jednačina se dodaje K jednačina tipa jednakosti:

$$\frac{\partial F(x, \lambda)}{\partial x_n} = 0, n=1,\ldots,N$$

$$h_k(x) = 0, k = 1, \ldots, K$$

Rešavanjem ovog sistema jednačina dobija se stacionarna tačka Lagranžove funkcije F. Izračunavanjem Hesijana funkcije F po promenljivoj X može se videti da li dobijena stacionarna tačka predstavlja globalni optimum.

2.2. Kun-Takerovi uslovi

Lagranžovi multiplikatori se koriste za razvoj uslova optimalnosti za optimizacione probleme sa ograničenja tipa jednakosti. Kun i Taker dali su proširenje na opšti nelinearni optimizacioni problem sa ograničenjima tipa jednakosti i nejednakosti:

$$\min f(x),$$

$$g_j(x) \geq 0, j = 1,2,...,J$$

$$h_k(x) = 0, k = 1,2, \ldots, K$$

$$x = [x_1, x_2, \ldots, x_N]^T$$

Za ograničenje tipa nejednakosti se kaže da je aktivno u x_0, ako je $g_j(x_0) = 0$. Inače se za ograničenje kaže da je neaktivno ako je $g_j(x_0) > 0$. Ako se idetnifikuju neaktivna ograničenja pre rešavanja optimizacionog problema, onda se ona mogu isključiti iz razmatranja i time redukovati dimenzije problema.

Može se definisati proširena Lagranžova funkcija:

$$F(x, \lambda, \mu) = f(x) - \sum_{j=1}^{J} \mu_j g_j(x) - \sum_{k=1}^{K} \lambda_k h_k(x)$$

Lagranžovi multiplikatori λ_k odgovaraju ograničenjima tipa jednakosti, a μ_j odgovaraju ograničenjima tipa nejednakosti (Kun-Takerovi multiplikatori).

Kun i Taker razvili su potrebne i dovoljne uslove za opšti nelinearni optimizacioni problem uz pretpostavku da su funkcije f, g_j, h_k diferencijabilne. Ovi uslovi poznati su kao Kun-Takerovi uskovi ili Kun-Takerov problem. Kun-Takerovi uslovi svode se na određivanje vektora x, μ, λ, koji zadovoljavaju sledeće uslove:

$$\nabla f(x) - \sum_{j=1}^{J} \mu_j \nabla g_j(x) - \sum_{k=1}^{K} \lambda_k \nabla h_k(x) = 0 \tag{1}$$

$$g_j(x) \geq 0, j = 1,2,...,J \tag{2}$$

$$h_k(x) = 0, k = 1,2,...K \tag{3}$$

$$\mu_j g_j(x) = 0, j = 1,2,...,J \tag{4}$$

$$\mu_j \geq 0, j = 1,2,...,J \tag{5}$$

Uslov (4) naziva se uslovom komplementarnosti i odnosi se na stanje ograničenja. Ako je j-to ograničenje neaktivno, odnosno $g_j(x_0) > 0$, onda je $\mu_j = 0$ i važi da je $\mu_j g_j(x_0) = 0$. U suprotnom, ako je j-to ograničenje aktivno, odnosno $g_j(x_0) = 0$, onda nije neophodno da multiplikator μ_j bude jednak 0, pošto je već ispunjeno da je $\mu_j g_j(x_0) = 0$. Ovi uslovi su potrebni uslovi za postojanje optimalnog rešenja u tački x^*.

3. Metod kaznenih funkcija

Problem minimizacije funkcije sa ograničenjem koji se posmatra:

$$min f(x), x \in X$$

gde je X dopustiv skup rešenja.

Među najjednostavnije metodame za rešavanje ovog tipa problema spadaju metode kaznenih funkcija, koje ovakav problem svode na niz problema optimizacije bez ograničenja. Kod ovakvih postupaka, aproksimacija se postiže dodavanjem kaznenog člana funkciji cilja u delu domena koji ne zadovoljava data ograničenja (nedopustivo područje). Ovako se pogoršava vrednost tako modifikovane funkcije cilja što proces traženja prilikom optimizacije tera iz nedopustivog područja. Kako su ograničenja ugrađena u modifikovanu funkciju cilja, problem postaje bez ograničenja i za njegovo rešavanje primenjuju se metode optimizacije bez ograničenja.

Modifikovana funkcija cilja se u dopustivom području mora poklapati sa zadatom funkcijom cilja. Zato je važno da parametar koji određuje težinu kazne ima zanemarljiv uticaj na iznos funkcije cilja na dopustivom skupu. Međutim, u delu domena koji ne zadovoljava ograničenja, funkcija cilja treba da ima velike vrednosti. To se postiže dodavanjem kaznenog člana, koji je definisan na takav način. Kako se funkcija cilja modifikuje, moguće je primeniti metode optimizacije bez ograničenja, a da ograničenja budu zadovoljena.

Prostoje dve vrste kaznenih funkcija, spoljašnje i unutrašnje kaznene funkcije.

Kako je pri rešavanju problema korišćen metod spoljašnjih kaznenih funkcija, on će biti opisana u nastavku.

3.1. Metod spoljašnjih kaznenih funkcija

Osnovna ideja kod metoda kaznenih funkcija je svođenje problema minimizacije na niz pomoćnih problema tako da se može približno rešiti rešavanjem niza problema optimizacije bez ogranicenja oblika:

$$minF_k(x), x \in X_0, k = 1,2,...$$

gde je $X \subset X_0 \subseteq R^n$.

Funkcije $F_k(x)$ su pomoćne funkcije, koje se određuju tako da se sa porastom k sve manje razlikuju od funkcije cilja $f(x)$ na skupu X, a van dopustivog skupa X trpe beskonačno veliku kaznu, što se postiže uvođenjem kaznenog parametra.

Skup X_0 je najčešće ceo prostor R^n ili neki njegov podskup. U ulozi skupa X_0 obično se uzima skup koji je zadat na jednostavan način tako da minimizacija pomoćnih funkcija na skupu X_0 ne predstavlja veliki problem.

Definicija: Kaznene funkcije skupa X na skupu X_0 ($X \subset X_0$) su funkcije $\{q_k(x)\}$, *k=1,2,...* koje su definisane i nenegativne na skupu X_0 sa osobinom:

$$\lim_{k\to\infty} q_k(x) = \begin{cases} 0, x \in X \\ +\infty, x \in X_0 \backslash X \end{cases}$$

Proširena funkcija $F_k(x)$ ima oblik:

$$F_k(x) = f(x) + q_k(x), x \in X_0, k = 1,2,...$$

Rešava se niz problema:

$$minF_k(x) = min(f(x) + q_k(x)), x \in X_0, k = 1,2,...$$

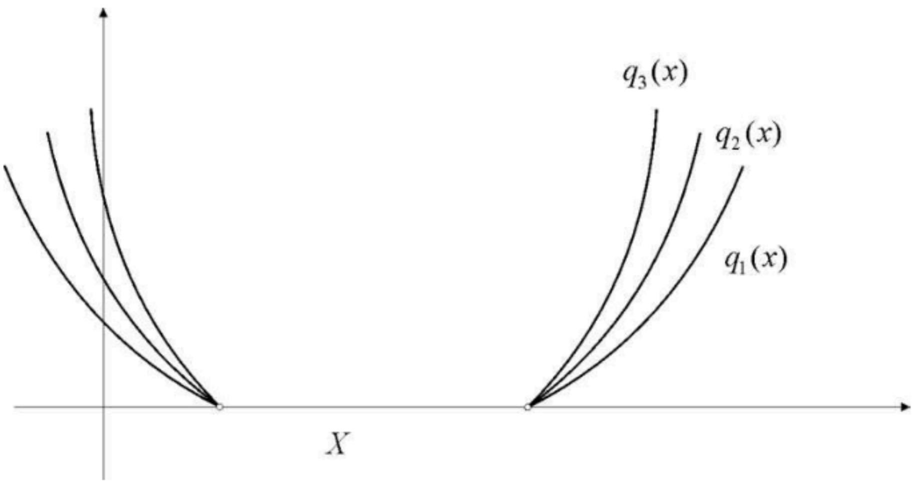
Rešenje problema minimizacije za fiksirano k se obeležava sa x^k, odnosno $F_k(x^k) = F_k$.

Za široku klasu problema važi $\lim_{k\to\infty} F_{k*} = f^*$ ili $\lim_{k\to\infty} x^k = x^*$, gde je $f^* = minf(x), x \in X$, a x^* je optimalno rešenje problema minimizacije.

Ukoliko se neki od problema ne mogu tačno rešiti, rešavaju se približno nekom metodom tako da važi:

$$F_k(x^k) \leq F_k + \epsilon_k \qquad \epsilon_k > 0, \lim_{k\to\infty} \epsilon_k = 0, k = 1,2,...$$

Geometrijska interpretacija jednog niza spoljašnjih kaznenih funkcija prikazana je na slici 1.



Slika 1. Niz spoljašnjih kaznenih funkcija

U dopustivom području, vrednosti ukupne modifikovane funkcije cilja su manje, jer tu nema kaznenog dodatka, zbog čega se ne stvara „zid" kako tok traženja ne bi izašao iz dopustivog područja, već kazneni dodatak „gura" tok traženja u dopustivo područje. Kod spoljašnjih kaznenih funkcija početna tačka i druge tačke u toku procesa traženja ne moraju biti u dopustivom području.

Oblik kaznenih funkcija osigurava porast modifikovane ciljne funkcije van dopustivog skupa, dok u dopustivom skupu nema kazne, odnosno kazneni dodatak je nula.

Neka je dopustiv skup X definisan na sledeći način:

$$X = \{x \in R^n, g_i(x) \leq 0, i = 1,...,m, g_i(x) = 0, i = m+1,...,s\}$$

Jedna od najčešće korišćenih spoljašnjih kaznenih funkcija za skup X je:

$$q_k(x) = t_k q(x)$$

gde je:

$$q(x) = \sum_{i=1}^{m} max\{0, g_i(x)^p\} + \sum_{i=m+1}^{s} |g_i(x)|^p, p \geq 1, x \in X_0$$

ili

$$q(x) = \sum_{i=1}^{s} (g_i(x) + |g_i(x)|)^2/2$$

gde su tk > 0, k=1,2,.. sa osobinom

$$\lim_{k \to \infty} t_k = +\infty$$

i zovu se kaznene konstante ili kazneni parametri.

Metoda spoljašnjih kaznenih funkcija implementira se sledećim algoritmom:

Korak 1. Neka je $\{t_k\}, k = 1,2,...$ niz koji teži beskonačnosti, tako da je svako $k, t_k \geq 0, t_{k+1} > t_k$

Korak 2. Za svako $k = 1,2,...$ definiše se funkcija

$$F_k(x) = f(x) + q_k(x)$$

i rešava se problem

$$min F_k(x), x \in X_0$$

Odakle se dobija rešenje x^k. U koraku 2 se pretpostavlja da za svako k problem minimizacije ima rešenje.

4. Rešenje zadatka

Problem minimizacije koji se rešava predstavljen je na sledeći način:

$$\min f(x), x \in S$$

gde je:

$$f(x) = (x-1)^2 + (y+2)^4$$

a skup S je opisan nejednakostima:

$$(x+1)^2 + 2(y+3)^2 \leq 3$$

$$y^2 \leq 6 - 5x$$

Za rešavanje problema korišćen je metod spoljašnjih kaznenih funkcija i Huk-Dživsov metod.

Na slici 2. data je implementacija Huk-Dživsovog metoda. Implementirana je i pomoćna funkcija, koja vrši proveru u „plus" i „minus" smeru kako bi kod bio pregledniji.

```
from sympy.abc import t, p, x, y
from sympy import simplify
from math import sqrt, pow
import numpy as np
from matplotlib import cm
import matplotlib.pyplot as plt


def check_in_d_direction(f, y0, delta, d):
    y1 = []

    if f.evalf(8, subs={x: y0[0], y: y0[1]}) > f.evalf(8, subs={x: y0[0] + delta * d[0], y: y0[1] + delta * d[1]}):
        y1 = [y0[0] + delta * d[0], y0[1] + delta * d[1]]
        # print("smer: plus, y=", y1, " f=", f.evalf(7, subs={x: y1[0], y: y1[1]}))
    elif f.evalf(8, subs={x: y0[0], y: y0[1]}) > f.evalf(8, subs={x: y0[0] - delta * d[0], y: y0[1] - delta * d[1]}):
        y1 = [y0[0] - delta * d[0], y0[1] - delta * d[1]]
        # print("smer: minus, y=", y1, " f=", f.evalf(7, subs={x: y1[0], y: y1[1]}))
    else:
        y1 = y0
    return y1


def hooke_jeeves(f, x0, delta, epsilon, d):
    br_iter = 0
    max_iter = 50
    while delta >= epsilon and br_iter < max_iter:
        br_iter = br_iter + 1

        #print("Iteracija broj: ", br_iter)
        y0 = x0
        y1 = y0
        for i in range(len(x0)):
            y1 = check_in_d_direction(f, y0, delta, d[i])
            y0 = y1
        #print("Nova bazisna tacka x:", y1, "f=", f.evalf(7, subs={x: y1[0], y: y1[1]}), "delta=",delta)
        xxp = [y1[0] + (y1[0] - x0[0]), y1[1] + (y1[1] - x0[1])]
        if f.evalf(8, subs={x: y1[0], y: y1[1]}) > f.evalf(8, subs={x: xxp[0], y: xxp[1]}):
            y1 = xxp
            #print("\t Po obrascu: y=",y1," f=", f.evalf(7, subs={x: y1[0], y: y1[1]}))
        else:
            delta = delta * pow(10, -1)
            #print("\t Po obrascu: y=",xxp," f=", f.evalf(7, subs={x: xxp[0], y: xxp[1]}))
            #print("Povratak u staru bazisnu tacku sa manjim korakom")
        x0 = y1

    return x0
```

Slika 2. Huk-Dživsov metod i pomoćna funkcija

Na slici 3. prikazano je generisanje kaznenog parametra na osnovu zadatih ograničenja i implementacija metoda spoljašnjih kaznenih funkcija.

```
def exterior_penalty_function(constraints):
    penalty_function = ((constraints[0]+abs(constraints[0]))**2+(constraints[1]+abs(constraints[1]))**2)/2
    return simplify(penalty_function)


def exterior_penalty_minimization(f, constraints,x0, epsilon):
    penalty = exterior_penalty_function(constraints)
    ff = f + t * penalty
    print("Problem: min f(x,y), f(x,y) =",f)
    print("Ogranicenja: g1(x,y) =",constraints[0],"<= 0, g2(x,y) =", constraints[1],"<= 0")
    print("Pomocna funckija sa kaznom: F(x,y) = ",ff)
    t_value = 1
    max_iter = 50
    br_iter = 0
    difference = epsilon
    d = [[1, 0], [0, 1]]
    delta = 1
    points = []
    points.append(x0)
    while difference >= epsilon and br_iter <= max_iter:
        br_iter = br_iter + 1
        x1 = hooke_jeeves(ff.subs(t,t_value), x0, delta, epsilon, d)
        difference = sqrt(pow(x1[0]-x0[0],2) + pow(x1[1]-x0[1], 2))
        print("Iteracija: ", br_iter, end=", ")
        print("f=",f.evalf(9,subs={x:x1[0],y:x1[1]}), end=", ")
        print("{x,y}={",x1[0],",",x1[1],"}, t=", t_value)
        print("odstupanje=", difference,end=", ")
        print("vrednosti ogranicenja: {g1,g2}={",constraints[0].evalf(8,subs={x:x1[0],y:x1[1]}),end=",")
        print(constraints[1].evalf(8,subs={x:x1[0], y:x1[1]}),"}")
        t_value = t_value * 10
        x0 = x1
        points.append(x0)

    xmin = x0
    print("Resenje: f=",f.evalf(10,subs={x:xmin[0],y:xmin[1]}))
    print("Za {x,y}={",xmin[0],",",xmin[1],"}")

    return points
```

Slika 3. Generisanje kaznenog parametra i implementacija metoda spoljašnjih kaznenih funkcija

Na slici 4. priložen je kod glavnog programa, gde je definisan problem minimizacije i ograničenja, željena tačnost i početna tačka. Funkcija *plot_results* iscrtava niz tačaka dobijen kao rezultat izvršenja optimizacije.

```
if __name__ == '__main__':

    #definisanje problema i ogranicenja
    f = "(x-1)**2+(y+2)**4"
    g = [simplify("(x+1)**2+2*(y+3)**2-3"),simplify("y**2+5*x-6")]

    #definisanje zeljene tacnosti i pocetne tacke
    epsilon = 0.00000001
    x0 = [7, 4]
```

```
points = exterior_penalty_minimization(simplify(f), g, x0, epsilon)
```

```
plot_results(points)
```

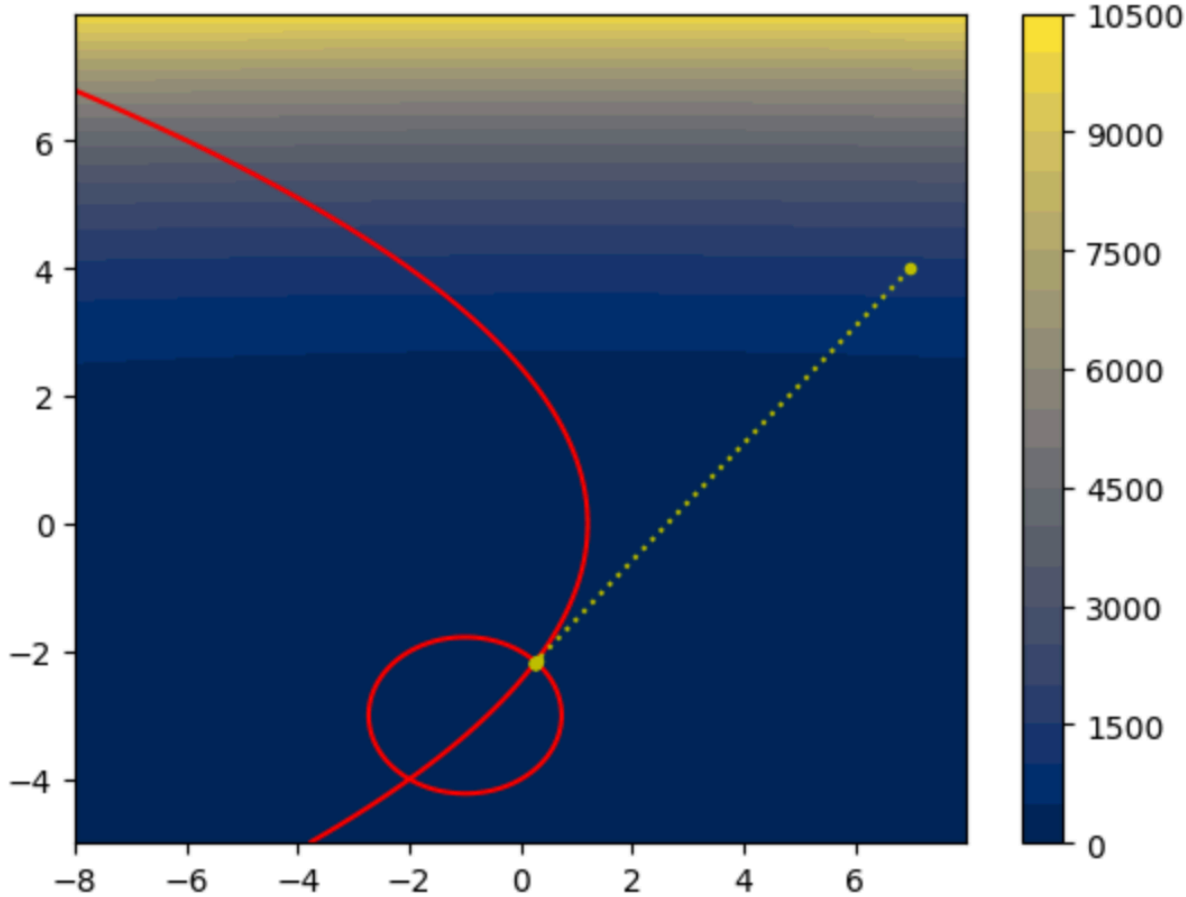
Slika 4. Definisanje problema minimizacije i poziv funkcije za minimizaciju i prikaz rezultata

Na slici 5. priložen je rezultat izvršenja funkcije *external_penalty_minimization*, gde se vidi vrednost funkcije f, ograničenja g i odstupanje u svakoj iteraciji, kao i tačka za koju se izračunava vrednost funkcije i ograničenja.

```
Iteracija:  1, f= 0.521353107, {x,y}={ 0.2783700000000001 , -2.156720000000001 }, t= 1
odstupanje= 9.115125397672816, vrednosti ogranicenja: {g1,g2}={ 0.056472174,0.043291158 }
Iteracija:  2, f= 0.539751870, {x,y}={ 0.2657939999999997 , -2.162273999999998 }, t= 10
odstupanje= 0.013747824991611462, vrednosti ogranicenja: {g1,g2}={ 0.0058041526,0.0043988511 }
Iteracija:  3, f= 0.541665124, {x,y}={ 0.26449899999999976 , -2.162854999999997 }, t= 100
odstupanje= 0.0014193611238859309, vrednosti ogranicenja: {g1,g2}={ 0.00058122305,0.00043675103 }
Iteracija:  4, f= 0.541856355, {x,y}={ 0.2643696999999999 , -2.1629135999999973 }, t= 1000
odstupanje= 0.00014195932516050279, vrednosti ogranicenja: {g1,g2}={ 5.8020408e-5,4.3741065e-5 }
Iteracija:  5, f= 0.541875554, {x,y}={ 0.26435671999999966 , -2.162919499999997 }, t= 10000
odstupanje= 1.4257994248903357e-05, vrednosti ogranicenja: {g1,g2}={ 5.4423697e-6,4.3634802e-6 }
Iteracija:  6, f= 0.541877387, {x,y}={ 0.2643554799999997 , -2.162920019999998 }, t= 100000
odstupanje= 1.3446189054024464e-06, vrednosti ogranicenja: {g1,g2}={ 5.6563964e-7,4.1291679e-7 }
Iteracija:  7, f= 0.541877594, {x,y}={ 0.26435533999999966 , -2.162920089999998 }, t= 1000000
odstupanje= 1.5652475846672426e-07, vrednosti ogranicenja: {g1,g2}={ -2.2762262e-8,1.5725597e-8 }
Iteracija:  8, f= 0.541877594, {x,y}={ 0.26435533999999966 , -2.162920079999998 }, t= 10000000
odstupanje= 9.99999993922529e-09, vrednosti ogranicenja: {g1,g2}={ 1.0720935e-8,-2.7532804e-8 }
Resenje: f= 0.5418775941
Za {x,y}={ 0.26435533999999966 , -2.162920079999998 }
```

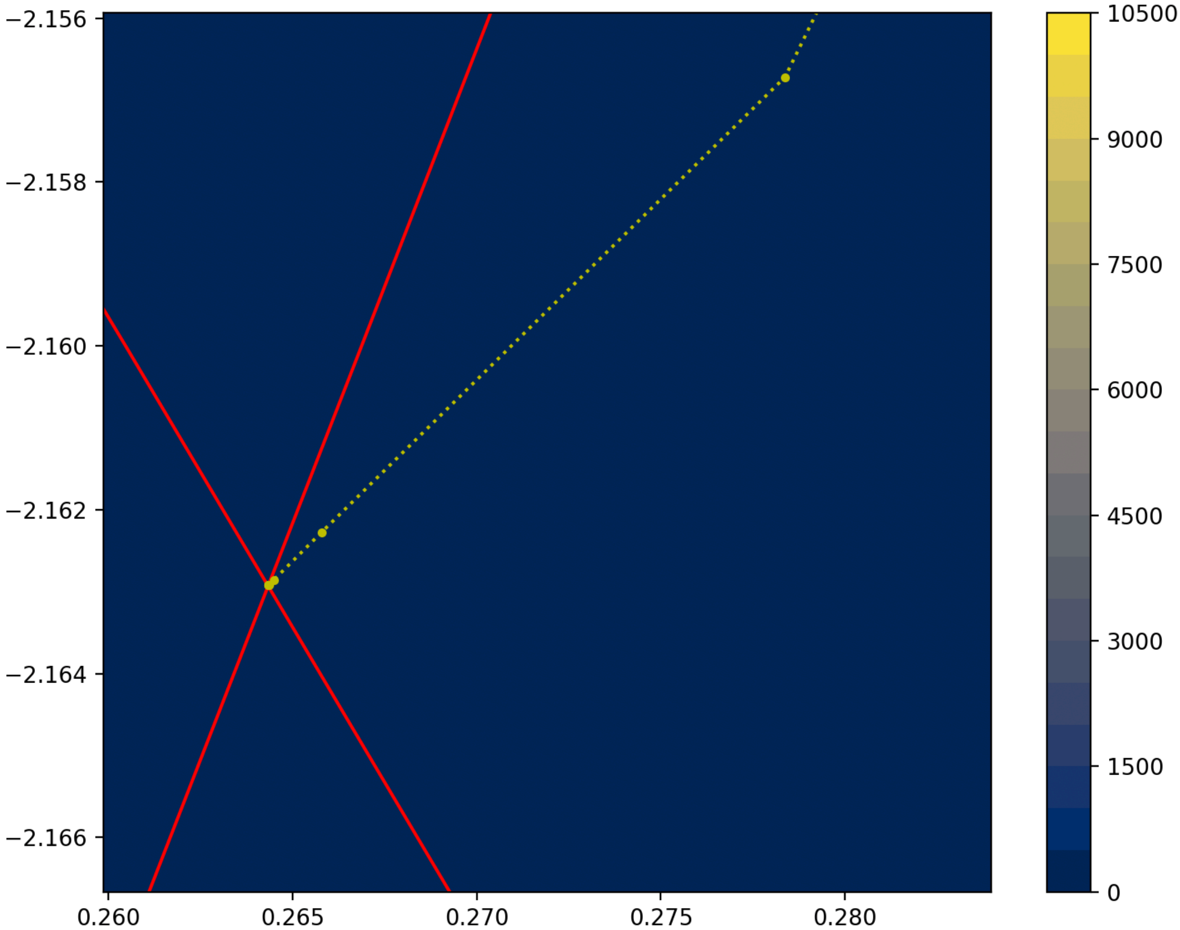
Slika 5. Rezultat minimizacije

Na slici 6. vizuelno je prikazna aproksimacija rešenja. Crvenim linijama označena su ograničenja. Početna tačka je $X_0 = [7,4]$. Može se primetiti da se već u prvoj iteraciji dolazi do približnog rešenja, a broj potrebnih iteracija zavisi od željene tačnosti, koja je u konkretnom primeru $\epsilon = 10^{-8}$.



Slika 6. Vizuelni prikaz minimizacije funkcije f

Ako se zumira grafikon u oblasti preseka ograničenja (slika 7) može se jasnije videti među-tačke, preko kojih se dolazi do rezultujuće tačke X_{min} = [0.26435533999999966 , -2.162920079999998], koja predstavlja zadovoljavajuće rešenje problema.



Slika 7. Zumiran prikaz aproksimacije tačke

5. Zaključak

U moderno doba ukazala se potreba da rad sistema, koji obavljaju određenu funkciju, bude efikasniji i ekonomičniji. Da bi sistem vršio funkciju na optimalan način, neophodno je fokusirati se na jedan cilj, što može biti dobit, efikasnost, smanjenje troškova i dr. Ako se cilj predstavi veličinom, odnosno ciljnom funkcijom, koja zavisi od nekih parametara, koji se u ciljnoj funkciji definišu promenljivim, optimizacijom se određuje vrednost tih parametara kako bi se ostvario definisani cilj. Pored ciljne funkcije i nezavisno promenljivih, definišu se i ograničenja, koja nezavisno promenljive moraju da zadovolje.

Kako su u praksi u većini slučajeva problemi optimizacije kompleksni, najčešće se sreću problemi optimizacije sa ograničenjima. Za rešavanje problema optimizacije sa ograničenjima mogu se koristiti kaznene funkcije.

U seminarskom radu data je teorijska osnova optimizacije sa ograničenjima, kao i metod kaznenih funkcija, koji je korišćen pri rešavanju zadatka, nakon čega je proloženo rešenje minimizacionog problema implementirano u *python* programskom jeziku, uz korišćenje biblioteka *sympy* i *matplotlib*.