KAUNO TECHNOLOGIJOS UNIVERSITETAS

Skaitiniai metodai ir algoritmai (P170B115)

Projektas

Antra užduotis

Atliko:

IFF-8/12 gr. studentas

Jokūbas Akramas

2020 m. lapkričio 29 d.

Priėmė:

lekt. Darius Naujokaitis

KAUNAS 2020

Turinys

1. Užduotis

Lygčių sistemų sprendimas (7 variantas)

1. Tiesinių lygčių sistemų sprendimas

Duota tiesinių lygčių sistema [A][X] = [B] ir jos sprendimui nurodytas metodas (1 lentelė).

1. Išspręskite tiesinių lygčių sistemą. Jeigu sprendinių be galo daug, raskite bent vieną iš jų. Jeigu sprendinių nėra, pagrįskite, kodėl taip yra.
   Jei metodas paremtas matricos pertvarkymu, pateikite matricų išraiškas kiekviename žingsnyje. Jei metodas iteracinis, grafiškai pavaizduokite, kaip atliekant iteracijas kinta santykinis sprendinio tikslumas esant kelioms skirtingoms konvergavimo daugiklio reikšmėms.
2. Patikrinkite gautus sprendinius ir skaidas, įrašydami juos į pradinę lygčių sistemą.
3. Gautą sprendinį patikrinkite naudodami išorinius išteklius (pvz., standartines MATLAB funkcijas).

2. Netiesinių lygčių sistemų sprendimas

1. Duota netiesinių lygčių sistema (2 lentelė. I lygčių sistema):
   $$\begin{cases} Z_1(x_1, x_2) = 0 \\ Z_2(x_1, x_2) = 0 \end{cases}$$

   a. Skirtinguose grafikuose pavaizduokite paviršius Z_1 (x_1, x_2) ir Z_2 (x_1, x_2).
   b. Užduotyje pateiktą netiesinių lygčių sistemą išspręskite grafiniu būdu.
   c. Užduotyje pateiktą netiesinių lygčių sistemą išspręskite naudodami užduotyje nurodytą metodą su laisvai pasirinktu pradiniu artiniu (išbandykite bent keturis pradinius artinius). Nurodykite iteracijų pabaigos sąlygas. Lentelėje pateikite pradinį artinį, tikslumą, iteracijų skaičių.
   d. Gautus sprendinius patikrinkite naudodami išorinius išteklius (pvz., standartines MATLAB funkcijas).

2. Duota netiesinių lygčių sistema (2 lentelė. II lygčių sistema):
   $$\begin{cases} Z_1(x_1,x_2,x_3,x_4) = 0 \\ Z_2(x_1,x_2,x_3,x_4) = 0 \\ Z_3(x_1,x_2,x_3,x_4) = 0 \\ Z_4(x_1,x_2,x_3,x_4) = 0 \end{cases}$$

   a. Užduotyje nurodytu metodu išspręskite netiesinių lygčių sistemą su laisvai pasirinktu pradiniu artiniu.
   b. Gautą sprendinį patikrinkite naudodami išorinius išteklius (pvz., standartines MATLAB funkcijas).

3. Optimizavimas

Pagal pateiktą uždavinio sąlygą (3 lentelė) sudarykite tikslo funkciją ir išspręskite jį vienu iš gradientinių metodų (gradientiniu, greičiausio nusileidimo, kvazi-gradientiniu, ar pan.). Gautą taškų konfigūraciją pavaizduokite programoje, skirtingais ženklais pavaizduokite duotus ir pridėtus (jei sąlygoje tokių yra) taškus. Ataskaitoje pateikite pradinę ir gautą taškų konfigūracijas, taikytos tikslo funkcijos aprašymą, taikyto metodo pavadinimą ir parametrus, iteracijų skaičių, iteracijų pabaigos sąlygas ir tikslo funkcijos priklausomybės nuo iteracijų skaičiaus grafiką.

1 lentelė. Tiesinių lygčių sistemų sprendimas. Užduotys.

Nr.	Lygčių sistema	Metodas	Nr.	Lygčių sistema	Metodas
1	$\begin{cases} 2x_1 + 5x_2 + x_3 + 2x_4 = 14 \\ -2x_1 + 3x_3 + 5x_4 = 10 \\ x_1 - x_3 + x_4 = 4 \\ 5x_2 + 4x_3 + 7x_4 = 24 \end{cases}$	Gauso	2	$\begin{cases} 3x_1 + 7x_2 + x_3 + 3x_4 = 37 \\ x_1 - 6x_2 + 6x_3 + 9x_4 = 11 \\ 4x_1 + 4x_2 - 7x_3 + x_4 = 38 \\ -x_1 + 3x_2 + 8x_3 + 2x_4 = -1 \end{cases}$	Gauso
3	$\begin{cases} 3x_1 + 10x_2 + x_3 + 5x_4 = 83 \\ -2x_1 + 6x_2 + 12x_3 + 14x_4 = 178 \\ 3x_1 + 12x_2 + 5x_3 + x_4 = 37 \\ -3x_1 - 9x_2 + 5x_3 = -26 \end{cases}$	Gauso – Žordano	4	$\begin{cases} 3x_1 + 7x_2 + x_3 + 3x_4 = 40 \\ x_1 - 6x_2 + 6x_3 + 8x_4 = 19 \\ 4x_1 + 4x_2 - 7x_3 + x_4 = 36 \\ 4x_1 + 16x_2 + 2x_3 = 48 \end{cases}$	Gauso – Žordano
5	$\begin{cases} x_1 + 2x_2 + x_3 + x_4 = -7 \\ 2x_1 - 5x_2 + x_3 + 2x_4 = 3 \\ 4x_1 - x_2 + 3x_3 + 4x_4 = 0 \\ 3x_1 - 3x_2 + 2x_3 + 3x_4 = 2 \end{cases}$	QR skaidos	6	$\begin{cases} 3x_1 + x_2 - x_3 + 5x_4 = 20 \\ -3x_1 + 4x_2 - 8x_3 - x_4 = -36 \\ x_1 - 3x_2 + 7x_3 + 6x_4 = 41 \\ 5x_2 - 9x_3 + 4x_4 = -16 \end{cases}$	QR skaidos
7	$\begin{cases} 9x_1 + x_2 - 2x_3 + x_4 = 47 \\ 11x_2 + 3x_3 + 4x_4 = -24 \\ x_1 + 3x_2 + 12x_3 - 3x_4 = 27 \\ -x_2 + 2x_3 + 2x_4 = -5 \end{cases}$	LU skaidos	8	$\begin{cases} 4x_1 + 12x_2 + x_3 + 7x_4 = 171 \\ 2x_1 + 6x_2 + 17x_3 + 2x_4 = 75 \\ 2x_1 + x_2 + 5x_3 + x_4 = 30 \\ 5x_1 + 11x_2 + 7x_3 = 50 \end{cases}$	Gauso

2 lentelė. Netiesinių lygčių sistemų sprendimas. Užduotys.

Nr.	I lygčių sistema	II lygčių sistema	Metodas
1	$\begin{cases} \sin(x_1)\cos(x_2) + \frac{x_2}{4} - 0.5 = 0 \\ e^{-3x_1^2 - x_2^2 + 3} - 0.1 = 0 \end{cases}$	$\begin{cases} 4x_2 + 3x_3 + 3x_4 - 26 = 0 \\ 3x_2 + 4x_2x_3 - 75 = 0 \\ x_3^3 - 2x_4^2 - 25 = 0 \\ 5x_1 - 12x_2 + 40 = 0 \end{cases}$	Broideno
2	$\begin{cases} 8\cos(x_1) + x_2^2 = 0 \\ 50e^{-\frac{x_1^2}{4} + x_2^2} + x_1 + x_2 - 5.5 = 0 \end{cases}$	$\begin{cases} x_2 - 2x_3 + 4x_4 + 5 = 0 \\ 4x_4^3 + 2x_2x_4 + 550 = 0 \\ 4x_3^3 - 2x_3^2 - 3x_1x_2 + 550 = 0 \\ 4x_1 - 3x_2 + 2x_3 + 2x_4 + 35 = 0 \end{cases}$	Greičiausio nusileidimo
3	$\begin{cases} \cos(x_1) - x_1 - x_2 = 0 \\ 20e^{-\frac{(x_1^2 + x_2^2)}{4}} + \frac{x_1^2 + x_2^2}{4} - 10 = 0 \end{cases}$	$\begin{cases} 2x_1 + 2x_2 - x_3 + 1 = 0 \\ -5x_4^2 + 4x_3x_4 - 4 = 0 \\ -3x_3^2 + x_4^3 - 2x_1x_4 + 3 = 0 \\ 3x_1 - 6x_2 + 2x_3 - 4x_4 + 44 = 0 \end{cases}$	Broideno
4	$\begin{cases} \left(\frac{x_1}{4}\right)^4 + \left(\frac{x_2}{4}\right)^4 - \left(\left(\frac{x_1}{2}\right)^2 + \left(\frac{x_2}{2}\right)^2\right) + 5 = 0 \\ x_1^2 + x_2^2 + x_1x_2 - 8(x_1 + x_2) - 4 = 0 \end{cases}$	$\begin{cases} 2x_2 + 4x_4 + 20 = 0 \\ x_1x_2 - x_4 - 14 = 0 \\ -3x_1^2 - x_2x_1 + 3x_4^3 + 277 = 0 \\ 3x_3 - 6x_2 + 2x_4 - 7 = 0 \end{cases}$	Greičiausio nusileidimo
5	$\begin{cases} x_2 \sin\left(\frac{x_1}{2}\right) - 0.1 = 0 \\ x_1^2 + \left(\frac{x_2}{4}\right)^4 - 12 = 0 \end{cases}$	$\begin{cases} 3x_1 + 4x_2 + 2x_3 + 3x_4 - 9 = 0 \\ 2x_3^2 - x_4^2 + 14 = 0 \\ 3x_1^2 + 3x_2^3 - 4x_4^2 - 14 = 0 \\ 4x_1 - 12x_2 - 8 = 0 \end{cases}$	Broideno
6	$\begin{cases} x_1(x_2 + 2\cos(x_1)) - 1 = 0 \\ x_1^4 + x_2^4 - 64 = 0 \end{cases}$	$\begin{cases} 3x_1 + 5x_2 + 3x_3 + x_4 - 8 = 0 \\ x_1^2 + 2x_2x_4 - 5 = 0 \\ -3x_2^2 - 3x_1x_2 + 2x_4^3 + 16 = 0 \\ 5x_1 - 15x_2 + 3x_4 + 22 = 0 \end{cases}$	Broideno
7	$\begin{cases} \frac{10x_1}{x_2^2 + 1} + x_1^2 - x_2^2 = 0 \\ x_1^2 + 2x_2^2 - 32 = 0 \end{cases}$	$\begin{cases} x_1 + 4x_2 + x_3 - 22 = 0 \\ x_2x_3 - 2x_3 - 18 = 0 \\ -x_2^2 + 2x_4^3 - 3x_1x_4 + 335 = 0 \\ 2x_3 - 12x_2 + 2x_4 + 58 = 0 \end{cases}$	Niutono

3 lentelė. Optimizavimo uždaviniai.

* **rekomenduojama** $\boldsymbol{n} \leq 20$, $\boldsymbol{m} \leq 20$ * (*papildymas*) Sudaryta struktūra turėtų būti analizuojama kaip pilnasis grafas, visų papildomai dedamų taškų pozicijos turėtų būti optimizuojamos vienu metu.
Uždavinys 1-6 variantams
Duotos **n** ($3 \leq \boldsymbol{n}$) taškų fiksuotos koordinatės ($-10 \leq \boldsymbol{x} \leq 10,\ -10 \leq \boldsymbol{y} \leq 10$). *(Koordinatės gali būti generuojamos atsitiktinai)*. Srityje ($-10 \leq \boldsymbol{x} \leq 10,\ -10 \leq \boldsymbol{y} \leq 10$) reikia padėti papildomų **m** ($3 \leq \boldsymbol{m}$) taškų taip, kad jų atstumai nuo visų kitų taškų (įskaitant ir papildomus) būtų kuo artimesni vidutiniam atstumui, o bendra taškų kaina kuo mažesnė. Vieno taško kaina apskaičiuojama pagal funkciją $C(x,y) = xe^{-\left(\frac{x^2+y^2}{10}\right)} + 1{,}5$.
Uždavinys 7-12 variantams
Plokštumoje ($-10 \leq \boldsymbol{x} \leq 10, -10 \leq \boldsymbol{y} \leq 10$) išsidėstę **n** taškų ($3 \leq \boldsymbol{n}$), vienas jų fiksuotas koordinačių pradžioje (0; 0). Kiekvienas taškas su visais kitais yra sujungtas tiesiomis linijomis (stygomis). Raskite tokias taškų koordinates, kad atstumas tarp taškų būtų kuo artimesnis vidutiniam atstumui, o stygų ilgių suma kuo geriau atitiktų nurodytą reikšmę **S** ($10 \leq \boldsymbol{S}$).

2. Rezultatai

1. Tiesinių lygčių sistemų sprendimas

Programos išvedamas rezultatas:

```
Sprendžiama lygčių sistema: [A][X]=[B]

[A] =
    9.000000    1.000000   -2.000000    1.000000
    0.000000   11.000000    3.000000    4.000000
    1.000000    3.000000   12.000000   -3.000000
    0.000000   -1.000000    2.000000    2.000000

[B] =
         47
        -24
         27
         -5
[L] =
    1.000000    0.000000    0.000000    0.000000
    0.000000    1.000000    0.000000    0.000000
    0.000000    0.000000    1.000000    0.000000
    0.000000    0.000000    0.000000    1.000000

[U] =
    0.000000    0.000000    0.000000    0.000000
    0.000000    0.000000    0.000000    0.000000
    0.000000    0.000000    0.000000    0.000000
    0.000000    0.000000    0.000000    0.000000

Nuliai po 1 stulp. Pertvarkyta matrica [A]=
    9.000000    1.000000   -2.000000    1.000000
    0.000000   11.000000    3.000000    4.000000
    0.000000    2.888889   12.222222   -3.111111
    0.000000   -1.000000    2.000000    2.000000

Nuliai po 2 stulp. Pertvarkyta matrica [A]=
    9.000000    1.000000   -2.000000    1.000000
    0.000000   11.000000    3.000000    4.000000
    0.000000    0.000000   11.434343   -4.161616
    0.000000    0.000000    2.272727    2.363636

Nuliai po 3 stulp. Pertvarkyta matrica [A]=
    9.000000    1.000000   -2.000000    1.000000
    0.000000   11.000000    3.000000    4.000000
    0.000000    0.000000   11.434343   -4.161616
    0.000000    0.000000    0.000000    3.190813

Skaičiavimai baigti. Rezultatai:

[L] =
    1.000000    0.000000    0.000000    0.000000
    0.000000    1.000000    0.000000    0.000000
    0.111111    0.262626    1.000000    0.000000
    0.000000   -0.090909    0.198763    1.000000

[U] =
    9.000000    1.000000   -2.000000    1.000000
    0.000000   11.000000    3.000000    4.000000
    0.000000    0.000000   11.434343   -4.161616
    0.000000    0.000000    0.000000    3.190813

Sprendinys [X] =
    6.027714
   -2.100451
    1.790986
   -1.566999

Tikrinimas =
1) [L]*[U] =
    9.000000    1.000000   -2.000000    1.000000
    0.000000   11.000000    3.000000    4.000000
    1.000000    3.000000   12.000000   -3.000000
    0.000000   -1.000000    2.000000    2.000000
```

```
2) Reikšmių įstatymas į pradinę matricą =
Tikrinama 1 pradinės matricos eilutė =
9.00 * 6.03 + 1.00 * -2.10 + -2.00 * 1.79 + 1.00 * -1.57 = 47.00. B[1] =
47.00
Tikrinama 2 pradinės matricos eilutė =
0.00 * 6.03 + 11.00 * -2.10 + 3.00 * 1.79 + 4.00 * -1.57 = -24.00. B[2] = -
24.00
Tikrinama 3 pradinės matricos eilutė =
1.00 * 6.03 + 3.00 * -2.10 + 12.00 * 1.79 + -3.00 * -1.57 = 25.92. B[3] =
27.00
Tikrinama 4 pradinės matricos eilutė =
0.00 * 6.03 + -1.00 * -2.10 + 2.00 * 1.79 + 2.00 * -1.57 = 2.55. B[4] = -
5.00
```

Sprendinių tikrinimas naudojant išorinius išteklius (MATLAB):

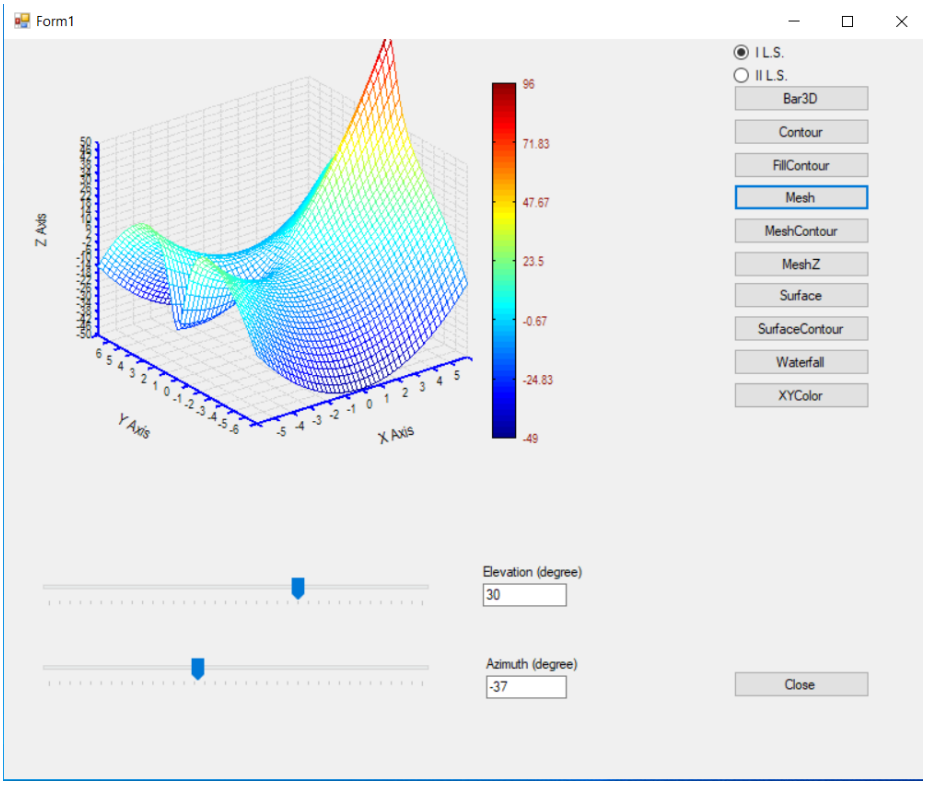
```
A =

     9     1    -2     1
     0    11     3     4
     1     3    12    -3
     0    -1     2     2


b =

    47
   -24
    27
    -5


n =

     4


L =

     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1


U =

     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0


U =

     9     1    -2     1
     0     0     0     0
     0     0     0     0
     0     0     0     0


A =

    9.0000    1.0000   -2.0000    1.0000
         0   11.0000    3.0000    4.0000
    0.1111    2.8889   12.2222   -3.1111
         0   -1.0000    2.0000    2.0000
```

```
A =

    9.0000    1.0000   -2.0000    1.0000
         0   11.0000    3.0000    4.0000
    0.1111    0.2626   11.4343   -4.1616
         0   -0.0909    2.2727    2.3636


A =

    9.0000    1.0000   -2.0000    1.0000
         0   11.0000    3.0000    4.0000
    0.1111    0.2626   11.4343   -4.1616
         0   -0.0909    0.1988    3.1908


U =

    9.0000    1.0000   -2.0000    1.0000
         0   11.0000    3.0000    4.0000
         0    0.0000   11.4343   -4.1616
         0         0         0    3.1908


L =

    1.0000         0         0         0
         0    1.0000         0         0
    0.1111    0.2626    1.0000         0
         0   -0.0909    0.1988    1.0000


ans =

    9.0000    1.0000   -2.0000    1.0000
         0   11.0000    3.0000    4.0000
    1.0000    3.0000   12.0000   -3.0000
         0   -1.0000    2.0000    2.0000

>>
```

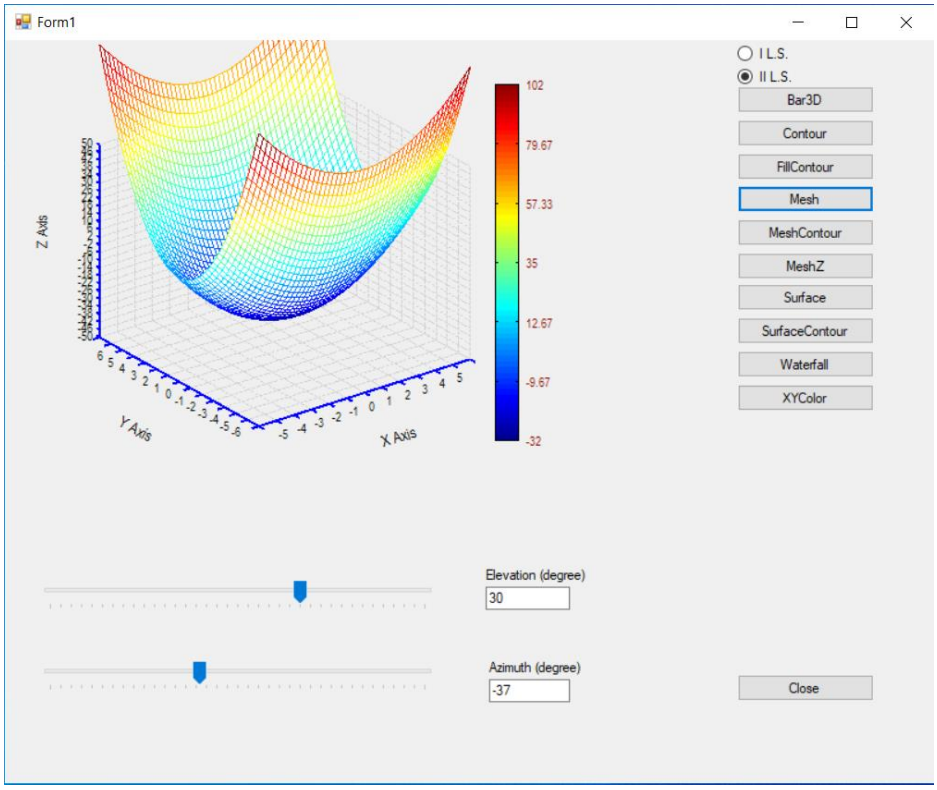
2. Netiesinių lygčių sistemų sprendimas

I lygčių sistema

Paviršių Z_1 (x_1, x_2) ir Z_2 (x_1, x_2) vaizdavimas skirtinguose grafikuose (pav. 1, 2)

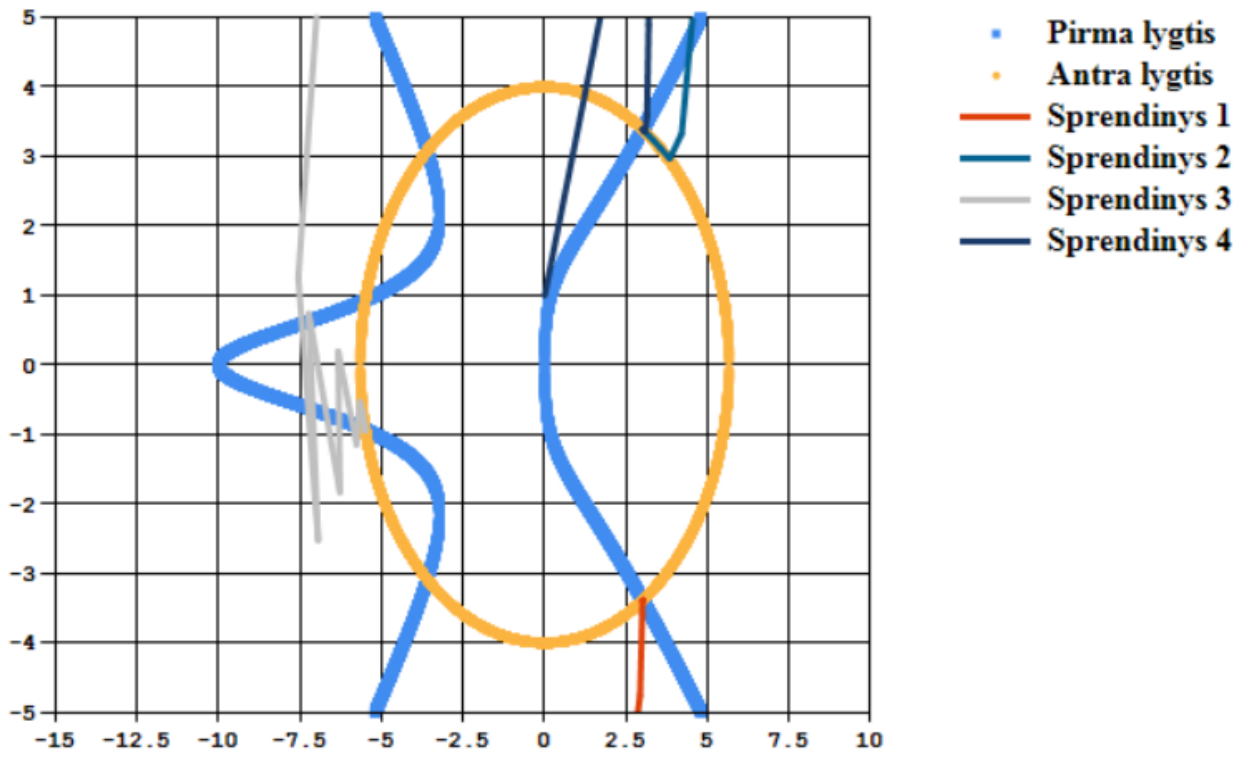


pav. 1 Paviršius Z1(x1, x2)



pav. 2 Paviršius Z2(x1, x2)

Užduoties grafinis sprendinys (pav. 3)



pav. 3 Grafinis sprendinys

Sprendimas Niutono metodu su keturiais pradiniais artiniais (programos išvedamas tekstas):

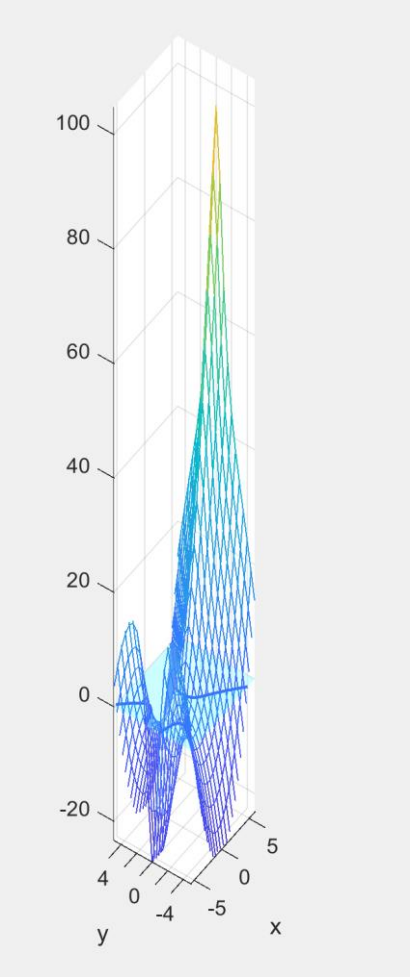
```
Pradinis artinys: [2; -8], tikslumas: 0.00054, iteracijų sk.: 12
Sprendinys 1: [3.00957; -3.38692]
Pradinis artinys: [5; 10], tikslumas: 0.00078, iteracijų sk.: 20
Sprendinys 2: [3.00970; 3.38687]
Pradinis artinys: [-7; 5], tikslumas: 0.00054, iteracijų sk.: 22
Sprendinys 3: [-5.50056; -0.93376]
Pradinis artinys: [0; 1], tikslumas: 0.00090, iteracijų sk.: 16
Sprendinys 4: [3.00971; 3.38686]
```

Iteracijos pabaigos sąlyga – kai tikslumas tampa mažesnis už 0.001

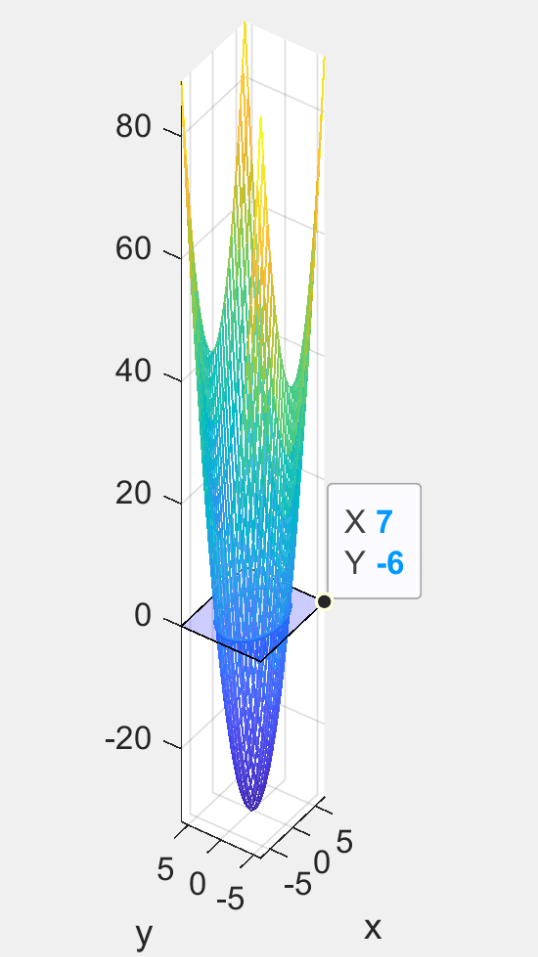
```
double tikslumas = Double.MaxValue;
while (tikslumas > 1e-3)
{
```

Sprendinių tikrinimas naudojant išorinius išteklius (MATLAB):

Grafinis paviršių vaizdavimas (pav. 4, 5)

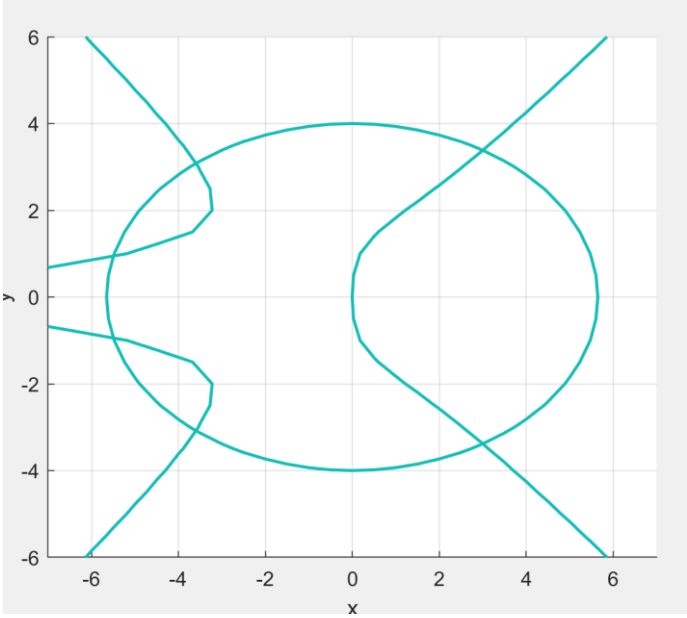


pav. 4 Paviršius Z1(x1,x2) (MATLAB)



pav. 5 Paviršius Z2(x1, x2) (MATLAB)

Grafinis netiesinių lygčių sistemos sprendimas (pav. 6)



pav. 6 Grafinis NLS sprendimas (MATLAB)

Sprendimas Niutono metodu su laisvai pasirinktu pradiniu artiniu (programos išvedamas tekstas):

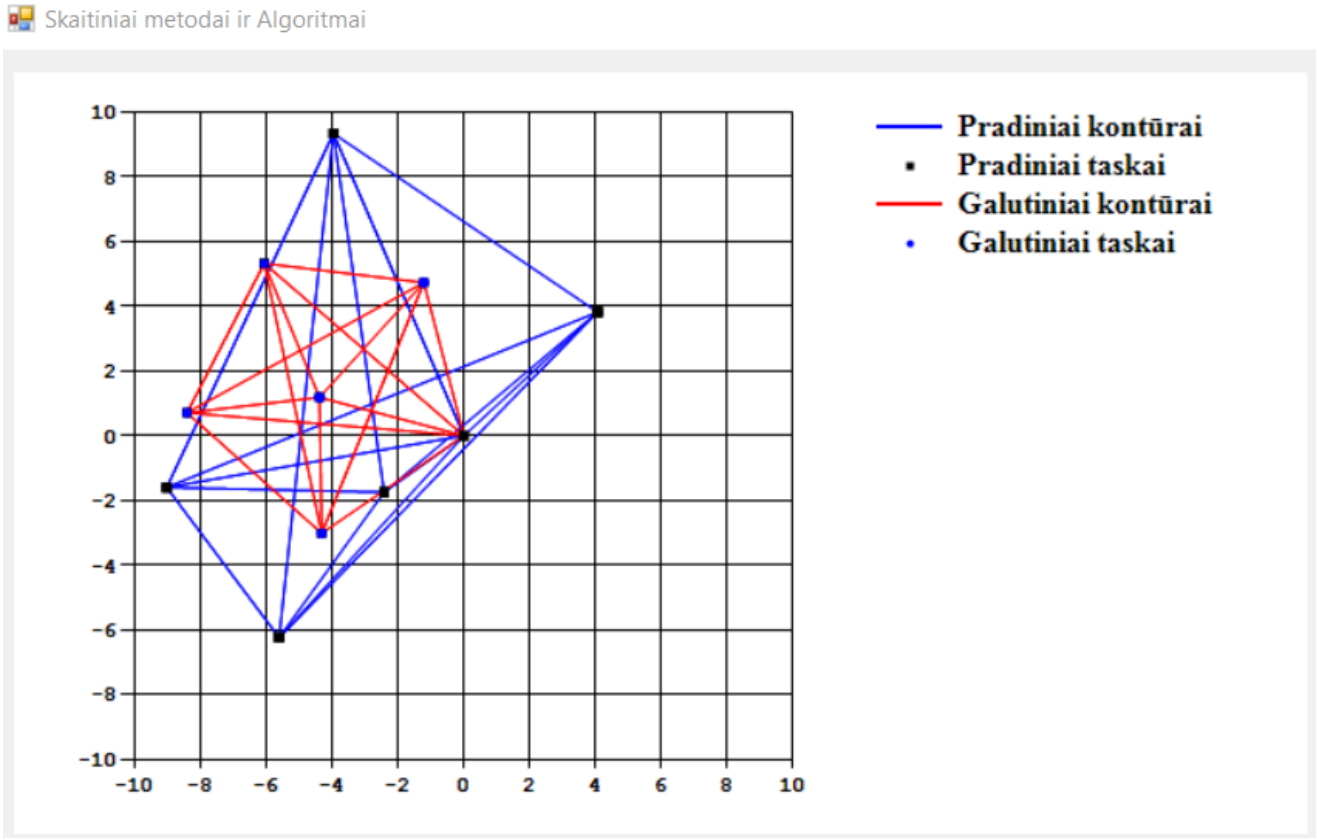
```
Artinys: [73.50000; -6.50000; -25.50000; -42.50000], tikslumas: 352.00000, iteracija : 1
Artinys: [45.97782; -2.36017; -14.53713; -28.62390], tikslumas: 144066.00000, iteracija : 2
Artinys: [34.17488; -0.26673; -11.10798; -19.49237], tikslumas: 42672.48635, iteracija : 3
Artinys: [31.74833; 0.55560; -11.97071; -13.69571], tikslumas: 12486.15981, iteracija : 4
Artinys: [33.09523; 0.75518; -14.11594; -10.35301], tikslumas: 3498.74045, iteracija : 5
Artinys: [33.94032; 0.82345; -15.23413; -8.82514], tikslumas: 857.03469, iteracija : 6
Artinys: [34.15057; 0.83970; -15.50936; -8.45244], tikslumas: 141.75229, iteracija : 7
Artinys: [34.16319; 0.84063; -15.52572; -8.43047], tikslumas: 7.48685, iteracija : 8
Artinys: [34.16323; 0.84064; -15.52578; -8.43039], tikslumas: 0.02530, iteracija : 9
Artinys: [34.16323; 0.84064; -15.52578; -8.43039], tikslumas: 0.00000, iteracija : 10
Pradinis artinys: [1; 1; 1; 1], tikslumas: 0.00000029, iteracijų sk.: 10
Sprendinys:  [34.16323; 0.84064; -15.52578; -8.43039]
```

Sprendinių tikrinimas naudojant išorinius išteklius (MATLAB):

```
 iteracija 1  tikslumas 0.500095
 iteracija 2  tikslumas 0.370082
 iteracija 3  tikslumas 0.274323
 iteracija 4  tikslumas 0.14878
 iteracija 5  tikslumas 0.10082
 iteracija 6  tikslumas 0.0514533
 iteracija 7  tikslumas 0.0130637
 iteracija 8  tikslumas 0.000783814
 iteracija 9  tikslumas 2.67046e-06
 iteracija 10  tikslumas 3.086e-11
 sprendinys x =  34.1632  0.840638  -15.5258  -8.43039
 funkcijos reiksme f =  -7.10543e-15  0  -1.13687e-13  0>>
```

3. Optimizavimas

Pradinė ir galutinė taškų konfigūracijos (pav. 7)



pav. 7 Taškų konfigūracijos

Sprendimas gradientiniu metodu (programos išvedamas tekstas):

```
Iteracija: 0, tikslumas: 0.02213756357277973, tikslo f-ija: 545.53
Iteracija: 1, tikslumas: 0.00415231547696496, tikslo f-ija: 541.00
Iteracija: 2, tikslumas: 0.00780412949742464, tikslo f-ija: 532.57
Iteracija: 3, tikslumas: 0.013541509174676, tikslo f-ija: 518.12
Iteracija: 4, tikslumas: 0.0184594046238382, tikslo f-ija: 498.43
Iteracija: 5, tikslumas: 0.00209321457207836, tikslo f-ija: 492.29
Iteracija: 6, tikslumas: 0.08156855426315 43, tikslo f-ija: 555.33
Iteracija: 7, tikslumas: 0.0148256676690928, tikslo f-ija: 485.76
Iteracija: 8, tikslumas: 0.000671236400902059, tikslo f-ija: 470.94
Iteracija: 9, tikslumas: 0.00683261152534065, tikslo f-ija: 470.99
Iteracija: 10, tikslumas: 0.00102792838231593, tikslo f-ija: 463.64
Iteracija: 11, tikslumas: 0.00363537677959307, tikslo f-ija: 463.46
Iteracija: 12, tikslumas: 0.00103356632674074, tikslo f-ija: 459.15
Iteracija: 13, tikslumas: 0.00197586723556813, tikslo f-ija: 459.14
Iteracija: 14, tikslumas: 0.000704318552522634, tikslo f-ija: 456.68
Iteracija: 15, tikslumas: 0.0110565782914753, tikslo f-ija: 466.31
Iteracija: 16, tikslumas: 0.00154311467671771, tikslo f-ija: 457.53
Iteracija: 17, tikslumas: 0.000288426991644203, tikslo f-ija: 455.85
Iteracija: 18, tikslumas: 0.0038246850729563, tikslo f-ija: 459.35
Iteracija: 19, tikslumas: 0.000112205439840033, tikslo f-ija: 455.95
Iteracija: 20, tikslumas: 0.000380807746957256, tikslo f-ija: 455.50
Iteracija: 21, tikslumas: 8.16922474456524E-05, tikslo f-ija: 455.40
Iteracija: 22, tikslumas: 0.00557904843570715, tikslo f-ija: 460.38
Iteracija: 23, tikslumas: 0.00089465986611336, tikslo f-ija: 456.08
Iteracija: 24, tikslumas: 1.01332282252401E-05, tikslo f-ija: 455.26
Iteracija: 25, tikslumas: 0.000728636780560949, tikslo f-ija: 455.88
Iteracija: 26, tikslumas: 5.47766538051005E-05, tikslo f-ija: 455.16
Iteracija: 27, tikslumas: 0.00113630386864505, tikslo f-ija: 456.19
Iteracija: 28, tikslumas: 3.44998419083376E-05, tikslo f-ija: 455.19
Iteracija: 29, tikslumas: 0.000113309894042506, tikslo f-ija: 455.05
Iteracija: 30, tikslumas: 1.59982441455876E-05, tikslo f-ija: 455.03
```

```
Iteracija: 31, tikslumas: 0.000748925646758512, tikslo f-ija: 455.67
Iteracija: 32, tikslumas: 9.64272406988478E-05, tikslo f-ija: 455.07
Iteracija: 33, tikslumas: 2.03942170820039E-05, tikslo f-ija: 454.96
Iteracija: 34, tikslumas: 7.47293394420607E-05, tikslo f-ija: 455.02
Iteracija: 35, tikslumas: 2.57080552467774E-05, tikslo f-ija: 454.93
Iteracija: 36, tikslumas: 0.000117732870341974, tikslo f-ija: 455.04
Iteracija: 37, tikslumas: 1.76805447572454E-05, tikslo f-ija: 454.91
Iteracija: 38, tikslumas: 0.000104751217983336, tikslo f-ija: 455.00
Iteracija: 39, tikslumas: 2.57937772401267E-05, tikslo f-ija: 454.88
Iteracija: 40, tikslumas: 0.000148921633744215, tikslo f-ija: 455.02
Iteracija: 41, tikslumas: 1.6881511616427E-05, tikslo f-ija: 454.86
Iteracija: 42, tikslumas: 0.00012919860933808, tikslo f-ija: 454.98
Iteracija: 43, tikslumas: 2.64949083801705E-05, tikslo f-ija: 454.84
Iteracija: 44, tikslumas: 0.000171611165980482, tikslo f-ija: 454.99
Iteracija: 45, tikslumas: 1.52932465329635E-05, tikslo f-ija: 454.82
Iteracija: 46, tikslumas: 0.000149434359088575, tikslo f-ija: 454.95
Iteracija: 47, tikslumas: 2.57084802075631E-05, tikslo f-ija: 454.79
Iteracija: 48, tikslumas: 0.000193294393796976, tikslo f-ija: 454.97
Iteracija: 49, tikslumas: 1.26000043826963E-05, tikslo f-ija: 454.78
Iteracija: 50, tikslumas: 0.000169464737352955, tikslo f-ija: 454.93
Iteracija: 51, tikslumas: 2.40901623175568E-05, tikslo f-ija: 454.76
Iteracija: 52, tikslumas: 0.000215579666273084, tikslo f-ija: 454.95
Iteracija: 53, tikslumas: 9.2093828627691E-06, tikslo f-ija: 454.75
Iteracija: 54, tikslumas: 0.00018938420782863, tikslo f-ija: 454.92
Iteracija: 55, tikslumas: 2.21728142947537E-05, tikslo f-ija: 454.72
Iteracija: 56, tikslumas: 0.000237536264929388, tikslo f-ija: 454.94
Iteracija: 57, tikslumas: 5.58931761016653E-06, tikslo f-ija: 454.72
Iteracija: 58, tikslumas: 0.000208344131603001, tikslo f-ija: 454.91
Iteracija: 59, tikslumas: 2.02941037691902E-05, tikslo f-ija: 454.70
Iteracija: 60, tikslumas: 0.000257828023736862, tikslo f-ija: 454.93
Iteracija: 61, tikslumas: 2.14079101708395E-06, tikslo f-ija: 454.70
Iteracija: 62, tikslumas: 0.000225625552098242, tikslo f-ija: 454.90
Iteracija: 63, tikslumas: 1.85967384734021E-05, tikslo f-ija: 454.68
Iteracija: 64, tikslumas: 0.000275457736878582, tikslo f-ija: 454.93
Baigta sekmingai
Iteracijų skaičius = 65, tikslumas = 8.55911458899019E-07

Pradiniai taškai:
[(0.00, 0.00), (-3.97, 9.35), (-9.03, -1.60), (-5.61, -6.24), (4.06, 3.82), (-2.43, -1.75)]

Gauti taškai:
[(0.00, 0.00), (-6.07, 5.33), (-8.41, 0.71), (-4.32, -3.02), (-1.22, 4.74), (-4.39, 1.18)]
```

Pradinė taškų konfigūracija:

```
[(0.00, 0.00), (-3.97, 9.35), (-9.03, -1.60), (-5.61, -6.24), (4.06, 3.82), (-2.43, -1.75)]
```

Gauta taškų konfigūracija:

```
[(0.00, 0.00), (-6.07, 5.33), (-8.41, 0.71), (-4.32, -3.02), (-1.22, 4.74), (-4.39, 1.18)]
```

Taikyta tikslo funkcija:

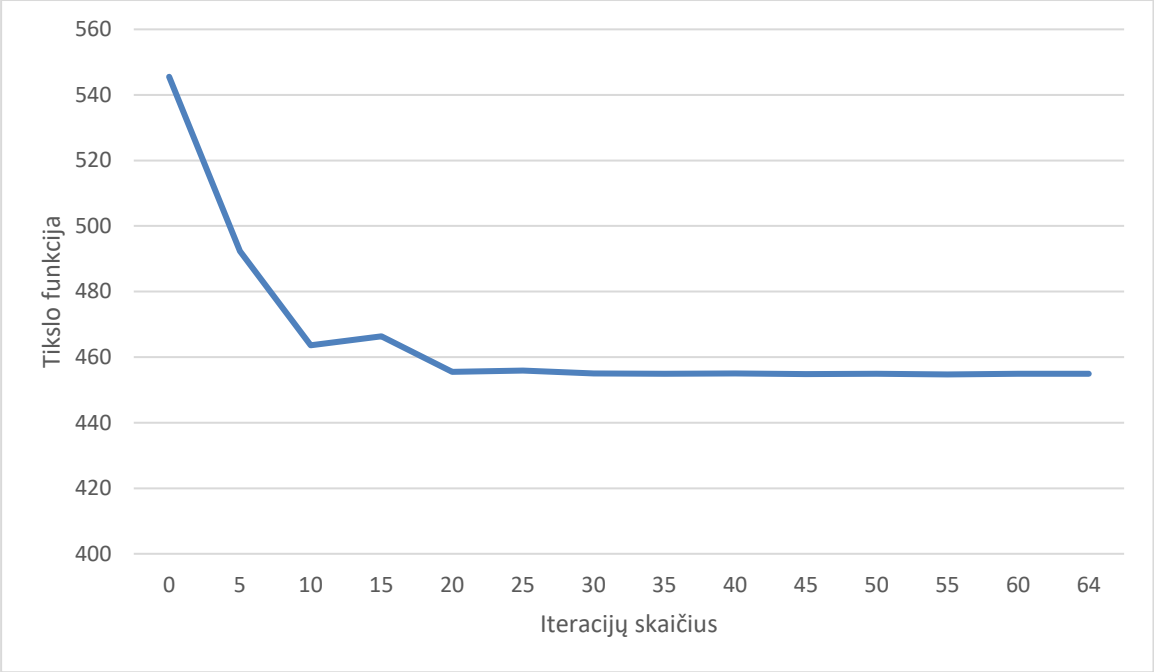
$$\Psi(\mathrm{x},\mathrm{y}) = \sum_{i=1}^{n}\sum_{u=i+1}^{n}\left(\sqrt{(x_u - x_i)^2 + (y_u - y_i)^2} - vid\right)^2 + \sum_{i=1}^{n}\sum_{u=i+1}^{n}\sqrt{(x_u - x_i)^2 + (y_u - y_i)^2} - s$$

Pirmasis funkcijos dėmuo yra kiekvieno taško atstumo su kiekvienu tašku atimtis iš vidutinio atstumo tarp taškų, pakelta kvadratu. T.y. kuo atstumai tarp taškų labiau atitiks *‚vid‘* kintamąjį (vidurkį), tuo tikslo funkcija bus mažesnė. Kėlimas kvadratu neatitikima vidurkiui didina eksponentiškai, kas tikslo funkciją daro didesnę, o tai papildomai atitolina funkciją nuo norimo tikslumo. Tokiu būdu pasiekiamas tikslesnis vidurkis.
Antrasis funkcijos dėmuo yra visų stygų suma, o argumentas *‚s‘* atimamas dėl tokios pat priežasties, kaip ir *‚vid‘* argumentas pirmame dėmenyje – tam, kad stygų sumai, labiausiai atitinkant argumentą *‚s‘* tikslo funkcija būtų kaip galima mažesnė.

Iteracijų skaičius: 65

Iteracijų pabaigos sąlyga: kada tikslumas bus mažesnis už 1e-6

Tikslo funkcijos priklausomybės nuo iteracijų skaičiaus grafikas



3. Programų kodai

1. Tiesinių lygčių sistemų sprendimas

```
1.  public static decimal[,] mulMatrix(decimal[,] a, decimal[,] b)
2.  {
3.      decimal[,] ret = new decimal[a.GetLength(1), b.GetLength(0)];
4.      for (int i = 0; i < ret.GetLength(0); i++)
5.      {
6.          for (int j = 0; j < ret.GetLength(1); j++)
7.          {
8.              ret[i, j] = 0;
9.              for (int k = 0; k < ret.GetLength(0); k++)
10.             {
11.                 ret[i, j] += a[i, k] * b[k, j];
12.             }
13.         }
14.     }
15.     return ret;
16. }
17. public static string printMatrix(decimal[,] A)
18. {
19.     StringBuilder ret = new StringBuilder();
20.     for (int i = 0; i < A.GetLength(0); i++)
21.     {
22.         for (int u = 0; u < A.GetLength(1); u++)
23.         {
24.             ret.Append(string.Format("{0, 12:F6}", A[i, u]));
25.         }
26.         ret.Append("\n");
27.     }
28.     ret.Append("\n");
29.     return ret.ToString();
30. }
31. private void button2_Click(object sender, EventArgs e)
32. {
33.     ClearForm1();
34.     richTextBox1.AppendText("Sprendžiama lygčių sistema: [A][X]=[B]\n\n");
35.     decimal[,] A = { { 9, 1, -2, 1 }, { 0, 11, 3, 4 }, { 1, 3, 12, -3 }, { 0, -1, 2, 2 } };
36.     decimal[,] A_test = { { 9, 1, -2, 1 }, { 0, 11, 3, 4 }, { 1, 3, 12, -3 }, { 0, -1, 2, 2 } };
37.     decimal[] B = { 47, -24, 27, -5 };
38.     decimal[] X = new decimal[B.Length];
39.     Array.ForEach(X, element => element = 0);
40.     richTextBox1.AppendText("[A] = \n");
41.     richTextBox1.AppendText(printMatrix(A));
42.     richTextBox1.AppendText("[B] = \n");
43.     Array.ForEach(B, element => richTextBox1.AppendText(string.Format("{0, 12}\n", element)));
44.     int n = A.GetLength(0);
45.     //---
46.     decimal[,] L = new decimal[n, n];
47.     decimal[,] U = new decimal[n, n];
48.     for (int i = 0; i < n; i++)
49.     {
50.         for (int u = 0; u < n; u++)
51.         {
52.             U[i, u] = 0;
53.             L[i, u] = (i == u) ? 1 : 0; //Vienetai po pagrindine įstrižaine
54.         }
55.     }
56.     //---
57.     richTextBox1.AppendText("[L] = \n");
58.     richTextBox1.AppendText(printMatrix(L));
59.     richTextBox1.AppendText("[U] = \n");
60.     richTextBox1.AppendText(printMatrix(U));
61.     //---
62.     for (int i = 0; i < n; i++) U[0, i] = A[0, i];
63.     decimal r = 0;
```

```
    for (int i = 0; i < n - 1; i++) //Eilute
    {
        for (int u = i + 1; u < n; u++) //Likusios eilutes
        {
            r = A[u, i] / A[i, i];
            for (int y = i; y < n; y++)
            {
                U[u, y] = A[u, y] - A[i, y] * r;
                A[u, y] = A[u, y] - A[i, y] * r;
            }
            L[u, i] = r;
        }
        richTextBox1.AppendText(string.Format("Nuliai po {0} stulp. Pertvarkyta matrica [A]=\n", i + 1));
        richTextBox1.AppendText(printMatrix(A));
    }
    //---
    richTextBox1.AppendText("Skaičiavimai baigti. Rezultatai:\n\n");
    richTextBox1.AppendText("[L] = \n");
    richTextBox1.AppendText(printMatrix(L));
    richTextBox1.AppendText("[U] = \n");
    richTextBox1.AppendText(printMatrix(U));
    //---
    for (int i = n - 1; i >= 0; i--)
    {
        decimal value = 0;
        for (int u = n - 1; u > i; u--)
        {
            value += U[i, u] * X[u];
        }
        X[i] = (B[i] - value) / U[i, i];
    }
    richTextBox1.AppendText("Sprendinys [X] = \n");
    Array.ForEach(X, element => richTextBox1.AppendText(string.Format("{0, 12:F6}\n", element)));
    //---
    //Tikrinimas
    richTextBox1.AppendText("\nTikrinimas = \n");
            //1. L*U
            decimal[,] ats = mulMatrix(L, U);
            richTextBox1.AppendText("1) [L]*[U] = \n");
            richTextBox1.AppendText(printMatrix(ats));
            //2. Reiksmiu istatymas
            richTextBox1.AppendText("2) Reikšmių įstatymas į pradinę matricą =\n");
            for (int i = 0; i < A_test.GetLength(0); i++)
            {
                richTextBox1.AppendText(string.Format("Tikrinama {0} pradinės matricos eilutė = \n", i + 1));
                decimal ats2 = 0;
                for (int u = 0; u < A_test.GetLength(1); u++)
                {
                    ats2 += A_test[i, u] * X[u];
                    richTextBox1.AppendText(string.Format("{0, 0:F2} * {1, 0:F2}", A_test[i, u], X[u]));
                    if (u + 1 < A_test.GetLength(1)) richTextBox1.AppendText(" + ");
                }
                richTextBox1.AppendText(string.Format(" = {0, 0:F2}. B[{1}] = {2, 0:F2}\n", ats2, i + 1, B[i]));
            }

        }
```

2. Netiesinių lygčių sistemų sprendimas

I lygčių sistema

```
1.  private double Y11(double x)
2.  {
3.      return Math.Sqrt(-((-
    Math.Pow(x, 2) + 1 + Math.Sqrt(Math.Pow(x, 4) + 2 * Math.Pow(x, 2) + 40 * x + 1)) / 2));
4.
5.  }
6.  private double Y12(double x)
7.  {
8.      return -Math.Sqrt(-((-
    Math.Pow(x, 2) + 1 + Math.Sqrt(Math.Pow(x, 4) + 2 * Math.Pow(x, 2) + 40 * x + 1)) / 2));
9.
10. }
11. private double Y13(double x)
12. {
13.     return Math.Sqrt(-((-
    Math.Pow(x, 2) + 1 - Math.Sqrt(Math.Pow(x, 4) + 2 * Math.Pow(x, 2) + 40 * x + 1)) / 2));
14.
15. }
16. private double Y14(double x)
17. {
18.     return -Math.Sqrt(-((-
    Math.Pow(x, 2) + 1 - Math.Sqrt(Math.Pow(x, 4) + 2 * Math.Pow(x, 2) + 40 * x + 1)) / 2));
19.
20. }
21. private double Y21(double x)
22. {
23.     return Math.Sqrt((-Math.Pow(x, 2) + 32) / 2);
24. }
25. private double Y22(double x)
26. {
27.     return -Math.Sqrt((-Math.Pow(x, 2) + 32) / 2);
28. }
29. private double Z1(double x, double y)
30. {
31.     return (10 * x) / (Math.Pow(y, 2) + 1) + Math.Pow(x, 2) - Math.Pow(y, 2);
32. }
33. Series z1, p1, z2, p2, line1, line2, line3, line4;
34. private void button3_Click(object sender, EventArgs e)
35. {
36.     if (var1.Checked) nlsPirmas();
37.     else if (var2.Checked) nlsAntras();
38. }
39.
40. private void nlsPirmas()
41. {
42.     PracticalGuideCharts.Form1 trimateForma = new PracticalGuideCharts.Form1();
43.     trimateForma.ShowDialog();
44.     ClearForm1();
45.     PrepareareForm(-15, 10, -5, 5);
46.     z1 = chart1.Series.Add("Pirma lygtis");
47.     z1.ChartType = SeriesChartType.Point;
48.     z2 = chart1.Series.Add("Antra lygtis");
49.     z2.ChartType = SeriesChartType.Point;
50.     for (double i = -15; i < 8; i += 0.001f)
51.     {
52.         z1.Points.AddXY(i, Y11(i));
53.         z1.Points.AddXY(i, Y12(i));
54.         z1.Points.AddXY(i, Y13(i));
55.         z1.Points.AddXY(i, Y14(i));
56.         z2.Points.AddXY(i, Y21(i));
57.         z2.Points.AddXY(i, Y22(i));
58.     }
59.     richTextBox1.AppendText("Nupiesta\n");
60.
```

```
61.     z1.BorderWidth = 1;
62.     z2.BorderWidth = 1;
63.
64.     line1 = chart1.Series.Add("Sprendinys 1");
65.     line2 = chart1.Series.Add("Sprendinys 2");
66.     line3 = chart1.Series.Add("Sprendinys 3");
67.     line4 = chart1.Series.Add("Sprendinys 4");
68.
69.     Niutono(2, -8, line1);
70.     Niutono(5, 10, line2);
71.     Niutono(-7, 5, line3);
72.     Niutono(0, 1, line4);
73. }
74.
75. private double f211(double x, double y)
76. {
77.     return 10 * x / (Math.Pow(y, 2) + 1) + Math.Pow(x, 2) - Math.Pow(y, 2);
78. }
79. private double dFx211(double x, double y)
80. {
81.     return 10 / (Math.Pow(y, 2) + 1) + 2 * Math.Pow(x, 2);
82. }
83. private double dFy211(double x, double y)
84. {
85.     return -20 * x * y / (Math.Pow((Math.Pow(y, 2) + 1), 2)) - 2 * y;
86. }
87. private double f212(double x, double y)
88. {
89.     return Math.Pow(x, 2) + 2 * Math.Pow(y, 2) - 32;
90. }
91. private double deltaFx212(double x, double y)
92. {
93.     return 2 * x;
94. }
95. private double dFy212(double x, double y)
96. {
97.     return 4 * y;
98. }
99.
100.        double[] pt = new double[2]; //Pradinis artinys
101.        double[,] J = new double[2, 2]; //Jakobio matrica
102.        double[] F = new double[2]; //Funkcijos reiksmes
103.        double[] deltaX = new double[2]; //Delta X vektorius
104.        int it;
105.        int count = 0;
106.
107.        private void Niutono(double x0_art, double x1_art, Series line)
108.        {
109.            it = 0;
110.            line.ChartType = SeriesChartType.Line;
111.            pt[0] = x0_art;
112.            pt[1] = x1_art;
113.
114.            double tikslumas = Double.MaxValue;
115.            while (tikslumas > 1e-3)
116.            {
117.
118.                J[0, 0] = dFx211(pt[0], pt[1]);
119.                J[0, 1] = dFy211(pt[0], pt[1]);
120.                J[1, 0] = deltaFx212(pt[0], pt[1]);
121.                J[1, 1] = dFy212(pt[0], pt[1]);
122.
123.                F[0] = f211(pt[0], pt[1]);
124.
125.                F[1] = f212(pt[0], pt[1]);
126.
127.                //Jakobio matrica sprendžiama gauso metodu
128.                double k = J[1, 0] / J[0, 0];
129.                J[1, 0] -= J[0, 0] * k;
130.                J[1, 1] -= J[0, 1] * k;
131.                F[1] -= F[0] * k;
132.
```

```
        deltaX[1] = -F[1] / J[1, 1];
                deltaX[0] = (-F[0] - deltaX[1] * J[0, 1]) / J[0, 0];

                line.Points.AddXY(pt[0], pt[1]);

                tikslumas = Math.Abs(f211(pt[0], pt[1]) - f212(pt[0], pt[1]));
                it++;

                if (tikslumas < 0.001 || it > 1000) break;
                else
                {
                    pt[0] += deltaX[0];
                    pt[1] += deltaX[1];
                }
            }

            richTextBox1.AppendText(string.Format("Pradinis artinys: [{0}; {1}], tikslumas: {2, 0:F5
    }, iteracijų sk.: {3}\nSprendinys {6}: [{4, 0:F5}; {5, 0:F5}]\n",
                x0_art, x1_art, tikslumas, it, pt[0], pt[1], ++count));
            line.BorderWidth = 3;

        }
```

II lygčių sistema

```
private double[] f221(double[] x)
{
    double[] ret = new double[4];
    ret[0] = x[0] + 4 * x[1] + x[2] - 22;
    ret[1] = x[1] * x[2] - 2 * x[2] - 18;
    ret[2] = -Math.Pow(x[1], 2) + 2 * Math.Pow(x[3], 3) - 3 * x[0] * x[3] + 335;
    ret[3] = 2 * x[2] - 12 * x[1] + 2 * x[3] + 58;

    return ret;
}
private double[,] df221(double[] x, double[] f)
{
    double[,] ret = new double[4, 5];
    ret[0, 0] = 1; ret[0, 1] = 4; ret[0, 2] = 1; ret[0, 3] = 0; ret[0, 4] = f[0];
    ret[1, 0] = 0; ret[1, 1] = x[2]; ret[1, 2] = x[1] - 2; ret[1, 3] = 0; ret[1, 4] = f[1];
    ret[2, 0] = -3 * x[3]; ret[2, 1] = -
2 * x[1]; ret[2, 2] = 0; ret[2, 3] = 6 * Math.Pow(x[3], 2) - 3 * x[0]; ret[2, 4] = f[2];
    ret[3, 0] = 0; ret[3, 1] = -12; ret[3, 2] = 2; ret[3, 3] = 2; ret[3, 4] = f[3];

    return ret;
}
static void Gausas(double[,] a, int n)
{
    int i, j, k = 0, c;

    for (i = 0; i < n; i++)
    {
        if (a[i, i] == 0)
        {
            c = 1;
            while ((i + c) < n && a[i + c, i] == 0)
                c++;
            if ((i + c) == n)
            {
                break;
            }
            for (j = i, k = 0; k <= n; k++)
            {
                double temp = a[j, k];
                a[j, k] = a[j + c, k];
                a[j + c, k] = temp;
            }
        }
```

```
43.
44.         for (j = 0; j < n; j++)
45.         {
46.             if (i != j)
47.             {
48.                 double p = a[j, i] / a[i, i];
49.
50.                 for (k = 0; k <= n; k++)
51.                     a[j, k] = a[j, k] - (a[i, k]) * p;
52.             }
53.         }
54.     }
55. }
56. private void nlsAntras()
57. {
58.     ClearForm1();
59.     double[] x = { 1, 1, 1, 1 };
60.     double[] x_prad = new double[x.GetLength(0)];
61.     Array.Copy(x, x_prad, x.GetLength(0));
62.     double[] deltaX = new double[4];
63.     int n = x.GetLength(0);
64.     int it = 0;
65.     double tikslumas = Double.MaxValue;
66.
67.     while (tikslumas > 1e-3)
68.     {
69.         double[] F = f221(x);
70.         double[,] J = df221(x, F);
71.         Gausas(J, 4);
72.
73.         for (int i = 0; i < n; i++)
74.         {
75.             deltaX[i] = -J[i, n] / J[i, i];
76.         }
77.
78.         tikslumas = Math.Abs(F.Max() - F.Min());
79.         it++;
80.
81.         x[0] += deltaX[0];
82.         x[1] += deltaX[1];
83.         x[2] += deltaX[2];
84.         x[3] += deltaX[3];
85.
86.         richTextBox1.AppendText(string.Format("Artinys: [{0, 0:F5}; {1, 0:F5}; {2, 0:F5}; {3, 0:F5}
    ], tikslumas: {4, 0:F5}, iteracija : {5}\n",
87.             x[0], x[1], x[2], x[3], tikslumas, it));
88.     }
89.     richTextBox1.AppendText(string.Format("Pradinis artinys: [{0}; {1}; {2}; {3}], tikslumas: {4, 0
    :F8}, iteracijų sk.: {5}\nSprendinys:  [{6, 0:F5}; {7, 0:F5}; {8, 0:F5}; {9, 0:F5}]\n",
90.         x_prad[0], x_prad[1], x_prad[2], x_prad[3], tikslumas, it, x[0], x[1], x[2], x[3]));
91. }
```

3. Optimizavimas

```
1.  private void button5_Click(object sender, EventArgs e)
2.  {
3.      ClearForm1();
4.      PrepareareForm(-10, 10, -10, 10);
5.      //---
6.      double s = 500;
7.      int count = 6;
8.      //---
9.      Random randNum = new Random();
10.     double[] x = Enumerable.Repeat(0, count).Select(i => randNum.NextDouble()*20-10).ToArray();
11.     double[] y = Enumerable.Repeat(0, count).Select(i => randNum.NextDouble()*20-10).ToArray();
12.     x[0] = 0;
13.     y[0] = 0;
14.     //---
15.     double[] x_copy = new double[x.Length];
16.     Array.Copy(x, x_copy, x.Length);
17.     double[] y_copy = new double[y.Length];
18.     Array.Copy(y, y_copy, y.Length);
19.     //---
20.     z1 = chart1.Series.Add("Pradiniai kontūrai");
21.     z1.ChartType = SeriesChartType.Line;
22.     z1.Color = Color.Blue;
23.     //---
24.     p1 = chart1.Series.Add("Pradiniai taskai");
25.     p1.ChartType = SeriesChartType.Point;
26.     p1.Color = Color.Black;
27.     //---
28.     z2 = chart1.Series.Add("Galutiniai kontūrai");
29.     z2.ChartType = SeriesChartType.Line;
30.     z2.Color = Color.Red;
31.     //---
32.     p2 = chart1.Series.Add("Galutiniai taskai");
33.     p2.ChartType = SeriesChartType.Point;
34.     p2.Color = Color.Blue;
35.     //---
36.     for (int i = 0; i < x.Length; i++)
37.     {
38.         p1.Points.AddXY(x[i], y[i]);
39.         for (int u = i+1; u < x.Length; u++)
40.         {
41.             z1.Points.AddXY(x[i], y[i]);
42.             z1.Points.AddXY(x[u], y[u]);
43.             p1.Points.AddXY(x[u], y[u]);
44.             z1.Points.AddXY(x[i], y[i]);
45.         }
46.     }
47.     z1.BorderWidth = 1;
48.     p1.BorderWidth = 3;
49.     p2.BorderWidth = 3;
50.     z2.BorderWidth = 1;
51.
52.     //Sprendimas
53.     optimizacija(x, y, s);
54.
55.     richTextBox1.AppendText("\nPradiniai taškai:\n");
56.     logPoints(x_copy, y_copy);
57.     richTextBox1.AppendText("\nGauti taškai:\n");
58.     logPoints(x, y);
59. }
60. private void logPoints(double[] ptX, double[] ptY)
61. {
62.     richTextBox1.AppendText("[");
63.     for (int i = 0; i < ptX.Length; i++)
64.     {
65.         richTextBox1.AppendText(string.Format("({0, 0:F2}, {1, 0:F2})", ptX[i], ptY[i]));
```

```
        if (i < ptX.Length - 1)
        {
            richTextBox1.AppendText(", ");
        }
    }
    richTextBox1.AppendText("]\n");
}
public void printPoints(double[] x, double[] y)
{
    z2.Points.Clear();
    p2.Points.Clear();
    for (int i = 0; i < x.Length; i++)
    {
        p1.Points.AddXY(x[i], y[i]);
        for (int u = i + 1; u < x.Length; u++)
        {
            z2.Points.AddXY(x[i], y[i]);
            z2.Points.AddXY(x[u], y[u]);
            p2.Points.AddXY(x[u], y[u]);
            z2.Points.AddXY(x[i], y[i]);
        }
    }
}

private void optimizacija(double[] x, double[] y, double s)
{
    double eps = 1e-6;
    int maxIter = 500;
    double zingsnis = 0.1;
    double tikslumas = Double.MaxValue;
    int iteracija = 0;

    for (; iteracija < maxIter; iteracija++)
    {
                //printPoints(x, y);
                double vid = vidurkis(x, y);
                int n = x.Length;
                double[,] grad = gradientas(x, y, vid, s);
                double f0 = tikslo(x, y, vid, s);
                double[,] deltaX = gradiento_norma(grad, zingsnis);
                for(int u = 1; u < n; u++)
                {
                    x[u] -= deltaX[u, 0];
                    y[u] -= deltaX[u, 1];
                }
                double f1 = tikslo(x, y, vid, s);
                if (f1 > f0)
                {
                    for (int u = 1; u < n; u++)
                    {
                        x[u] += deltaX[u, 0];
                        y[u] += deltaX[u, 1];
                    }
                    zingsnis /= 2;
                }
                else
                {
                    zingsnis *= 2;
                }
                tikslumas = Math.Abs(f0-f1)/(Math.Abs(f0)+Math.Abs(f1));
                if (tikslumas < eps)
                {
                    richTextBox1.AppendText("Baigta sekmingai\n");
                    break;
                }
                else if (iteracija == maxIter - 1)
                {
                    richTextBox1.AppendText("Baigta nesekmingai\n");
                }
```

```
                richTextBox1.AppendText(string.Format("Iteracija: {0}, tikslumas: {1}, tikslo f-
    ija: {2, 0:F2}\n", iteracija, tikslumas, f1));
            }
            richTextBox1.AppendText(string.Format("Iteracijų skaičius = {0}, tikslumas = {1}\n", ite
    racija, tikslumas));
            printPoints(x, y);
        }

        private double[,] gradiento_norma(double[,] gradientas, double zingsnis)
        {
            double suma = 0;
            for (int i = 0; i < gradientas.GetLength(0); i++)
            {
                for (int u = 0; u < gradientas.GetLength(1); u++)
                {
                    suma += Math.Pow(gradientas[i, u], 2);
                }
            }
            double normale = Math.Sqrt(suma);
            double[,] copy = new double[gradientas.GetLength(0), gradientas.GetLength(1)];
            for (int i = 0; i < gradientas.GetLength(0); i++)
            {
                for (int u = 0; u < gradientas.GetLength(1); u++)
                {
                    copy[i, u] = gradientas[i, u] / normale * zingsnis;
                }
            }
            return copy;
        }

        private double vidurkis(double[] x, double[] y)
        {
            double n = x.Length;
            double suma = 0;
            int count = 0;
            for (int i = 0; i < n; i++)
            {
                for (int u = i + 1; u < n; u++)
                {
                    suma += Math.Sqrt(Math.Pow(x[u]-x[i], 2) + Math.Pow(y[u] - y[i], 2));
                    count++;
                }
            }
            return suma / count;
        }
        private double ilgis(double[] x, double[] y)
        {
            double n = x.Length;
            double suma = 0;
            for (int i = 0; i < n; i++)
            {
                for (int u = i + 1; u < n; u++)
                {
                    suma += Math.Sqrt(Math.Pow(x[u] - x[i], 2) + Math.Pow(y[u] - y[i], 2));
                }
            }
            return suma;
        }

        private double[,] gradientas(double[] x, double[] y, double vid, double s)
        {
            int n = x.Length;
            double zingsnis = 0.0001;
            double[,] grad = new double[n, 2];
            double f0 = tikslo(x, y, vid, s);

```

```
199.            for (int i = 0; i < n; i++)
200.            {
201.                grad[i, 0] = (tikslo(f1(x, i, zingsnis), y, vid, s) - f0) / zingsnis;
202.                grad[i, 1] = (tikslo(x, f1(y, i, zingsnis), vid, s) - f0) / zingsnis;
203.            }
204.
205.            return grad;
206.        }
207.
208.        private double[] f1(double[] a, int i, double zing)
209.        {
210.            int n = a.Length;
211.            double[] copy = new double[n];
212.            Array.Copy(a, copy, n);
213.            copy[i] += zing;
214.            return copy;
215.        }
216.
217.        private double tikslo(double[] x, double[] y, double vid, double s)
218.        {
219.            int n = x.Length;
220.            double suma = 0;
221.            for (int i = 0; i < n; i++)
222.            {
223.                for (int u = i + 1; u < n; u++)
224.                {
225.                    suma += Math.Pow(Math.Sqrt(Math.Pow(x[u] - x[i], 2) + Math.Pow(y[u] - y[i], 2))
    - vid, 2);
226.                }
227.            }
228.            return suma + Math.Abs(ilgis(x, y) - s);
229.        }
```

4. Išvados

Darant šį darbą buvo įsisavinti TLS, NLS sprendimo būdai (LU skaidos, Niutono metodai). Optimizuojant trečią uždavinį buvo įsisavintas gradientinis metodas.