

INFORMATIKOS FAKULTETAS

**Skaitiniai metodai ir algoritmai** (P170B115)

Projektas

Antra užuotis

**Studentas**: Gediminas Milašius, IFF-6/2

Turinys

[1. Įvadas **Error! Bookmark not defined.**](#_Toc524958483)

[1.1. Pirmoji dalis **Error! Bookmark not defined.**](#_Toc524958484)

[1.2. Antroji dalis **Error! Bookmark not defined.**](#_Toc524958485)

[2. Pirmoji užduotis 5](#_Toc524958486)

[2.1. Pirmas punktas **Error! Bookmark not defined.**](#_Toc524958487)

[2.1.1 „Grubus“ įvertis 5](#_Toc524958488)

[2.1.2 „Tikslesnis“ įvertis **Error! Bookmark not defined.**](#_Toc524958489)

[2.1.3 Galutinis įvertis **Error! Bookmark not defined.**](#_Toc524958490)

[2.2. Antras punktas **Error! Bookmark not defined.**](#_Toc524958491)

[2.3. Trečias punktas **Error! Bookmark not defined.**](#_Toc524958492)

[2.4. Ketvirtas punktas **Error! Bookmark not defined.**](#_Toc524958493)

[2.4.1 Skenavimo metodas **Error! Bookmark not defined.**](#_Toc524958494)

[2.4.2 Stygų metodas **Error! Bookmark not defined.**](#_Toc524958495)

[2.4.3 Paprastųjų iteracijų metodas **Error! Bookmark not defined.**](#_Toc524958496)

[3. Antroji užduotis **Error! Bookmark not defined.**](#_Toc524958497)

[4. Išvados 10](#_Toc524958498)

# Metodiniai nurodymai

**Bendrieji reikalavimai namų darbams**

Ataskaitos keliamos į Moodle iki gynimo dienos. Ataskaitoje pateikiama užduotis, rezultatai, programų kodai. Visais atvejais atsiskaitymo metu galima naudotis namų užduotyje ir laboratorinių darbų metu nagrinėtomis programomis.

Gynimo metu studentas privalo paaiškinti bet kurią programos išeities teksto eilutę; jeigu to padaryti nesugeba, darbas vertinamas 0. Gynimo metu pateikiama darbo sutapties patikros ataskaita.

**Lygčių sistemų sprendimas.**

**Tiesinių lygčių sistemų sprendimas**

Duota tiesinių lygčių sistema 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. 3. Gautą sprendinį patikrinkite naudodami išorinius išteklius (pvz., standartines MATLAB funkcijas).

**Netiesinių lygčių sistemų sprendimas**

1. Duota netiesinių lygčių sistema (2 lentelė. I lygčių sistema):
2. Skirtinguose grafikuose pavaizduokite paviršius 𝑍1 (𝑥1, 𝑥2 ) ir 𝑍2 (𝑥1, 𝑥2 ).
3. Užduotyje pateiktą netiesinių lygčių sistemą išspręskite grafiniu būdu.
4. 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.
5. Gautus sprendinius patikrinkite naudodami išorinius išteklius (pvz., standartines MATLAB funkcijas).
6. Duota netiesinių lygčių sistema (2 lentelė. II lygčių sistema):
7. Užduotyje nurodytu metodu išspręskite netiesinių lygčių sistemą su laisvai pasirinktu pradiniu artiniu.
8. Gautą sprendinį patikrinkite naudodami išorinius išteklius (pvz., standartines MATLAB funkcijas).

**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

|  |  |  |
| --- | --- | --- |
| **Nr.** | **Lygčių sistema** | **Metodas** |
| 14 |  | Gauso |

2 lentelė

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr.** | **I Lygčių sistema** | **II Lygčių sistema** | **Metodas** |
| 14 |  |  | Broideno |

3 lentelė

|  |
| --- |
| **Uždavinys 13-18 variantams** |
| Duotos (3 ≤ 𝒏 ≤ 20) taškų koordinatės (−10 ≤ 𝒙 ≤ 10, −10 ≤ 𝒚 ≤ 10). (Koordinatės gali būti generuojamos atsitiktiniu būdu). Reikia padėti papildomų (3 ≤ 𝒎 ≤ 20, 𝒎 ≤ 𝒏) taškų taip, kad jų atstumai nuo (3 ≤ 𝒌 ≤ 5, 𝒌 < 𝒏 + 𝒎) artimiausių taškų (įskaitant ir papildomus) būtų kuo artimesni vidutiniam atstumui. |

# Užduoties atlikimas

## Tiesinių lygčių sistemų sprendimas

Pateiktoje užduotyje reikia Gauso metodu išspręsti lygčių sistemą.

Užduotį realizuojantis kodas:

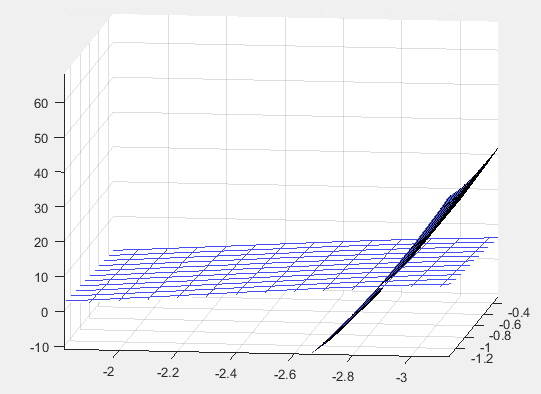
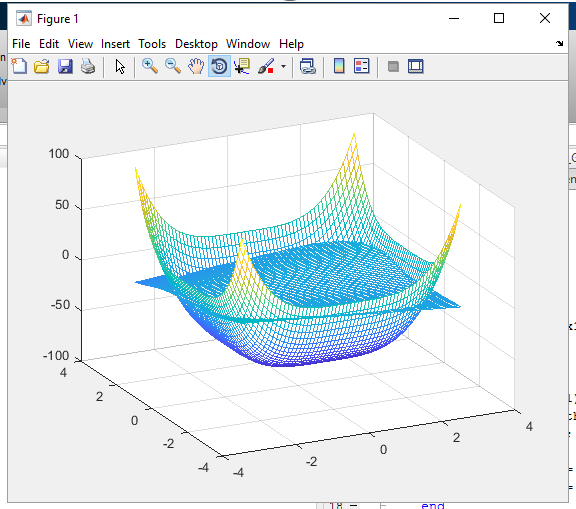
|  |
| --- |
| public void Run()  {  var eMatrix = new[] {  new double[] { 2, 5, 1, 2, 14 },  new double[] { -2, 0, 3, 5, 10 },  new double[] { 1, 0, -1, 1, 4 },  new double[] { 0, 5, 4, 7, 24 }  };  PrintMatrix(eMatrix);  for (int i = 0; i < eMatrix.Length - 1; i++)  {  FindAndSwapLeading(i, eMatrix);  Nullify(i, eMatrix);  \_form.OutputText("===================\n");  PrintMatrix(eMatrix);  }  \_form.OutputText("===================\n");  var res = Return(eMatrix);  if (res != null)  {  for (int i = 0; i < res.Length; i++)  {  \_form.OutputText($"x{i + 1} = {(Math.Sign(res[i]) >= 0 ? " " : "")}{res[i]:00.000}, ");  }  }  else  {  \_form.OutputText("Invalid setup. No possible solution found.");  return;  }  \_form.OutputText("\n===================\n");  for (int i = 0; i < res.Length; i++)  {  var val = 0.0;  for (int j = 0; j < res.Length; j++)  {  val += A[i][j] \* res[j];  \_form.OutputText($"{(Math.Sign(A[i][j]) >= 0 ? " " : "")}{A[i][j]:00.000} \* ");  \_form.OutputText($"{(Math.Sign(res[j]) >= 0 ? " " : "")}{res[j]:00.000} {(j < res.Length - 1 ? "+" : "")}");  }  \_form.OutputText($"= {val:00.000}\n");  }  }  private void Nullify(int diagonalIndex, double[][] matrix)  {  // Iterates through rows  for (int j = diagonalIndex + 1; j < matrix.Length; j++)  {  var multiplier = -matrix[j][diagonalIndex] / matrix[diagonalIndex][diagonalIndex];  // Iterates trough columns in row  for (int k = 0; k < matrix[j].Length; k++)  {  matrix[j][k] += multiplier \* matrix[diagonalIndex][k];  }  }  }  private static void FindAndSwapLeading(int i, double[][] matrix)  {  var hv = double.MinValue; var hi = 0; // Highest value, index  // Find hi, hv  for (int j = i; j < matrix.Length; j++)  {  if (Math.Abs(matrix[j][i]) > hv)  {  hv = Math.Abs(matrix[j][i]);  hi = j;  }  }  var tmp = matrix[i];  matrix[i] = matrix[hi];  matrix[hi] = tmp;  }  private double[] Return(double[][] matrix)  {  var result = new double[matrix.Length];  // Diagonal length  var dLength = matrix.Length;  for (int i = dLength - 1; i >= 0; i--)  {  result[i] = (matrix[i][dLength] - FindReturnResult(matrix, result, i)) / matrix[i][i];  if (double.IsNaN(result[i]))  {  // More than one result  \_form.OutputText($"{i + 1}: More than one result found. Assign x{i + 1} = 0.\n");  result[i] = 0;  }  else if (double.IsInfinity(result[i]))  {  // Result doesn't exist  return null;  }  }  return result;  }  private static double FindReturnResult(double[][] matrix, double[] answers, int index)  {  var result = 0.0;  for (int i = index; i < matrix.Length; i++)  {  result += matrix[index][i] \* answers[i];  }  return result;  }  private void PrintMatrix(double[][] matrix)  {  foreach (var row in matrix)  {  for (int j = 0; j < row.Length - 1; j++)  {  \_form.OutputText($"{(Math.Sign(row[j]) >= 0 ? " " : "")}{row[j]:00.000} ");  }  \_form.OutputText($"| {(Math.Sign(row[row.Length - 1]) >= 0 ? " " : "")}{row[row.Length - 1]:00.000}\n");  }  } |

**Matricų išraiškos kiekviename žingsnyje ir patikrinimas:**

|  |
| --- |
| 02.000 05.000 01.000 02.000 | 14.000  -02.000 00.000 03.000 05.000 | 10.000  01.000 00.000 -01.000 01.000 | 04.000  00.000 05.000 04.000 07.000 | 24.000  ===================  02.000 05.000 01.000 02.000 | 14.000  00.000 05.000 04.000 07.000 | 24.000  00.000 -02.500 -01.500 00.000 | -03.000  00.000 05.000 04.000 07.000 | 24.000  ===================  02.000 05.000 01.000 02.000 | 14.000  00.000 05.000 04.000 07.000 | 24.000  00.000 00.000 00.500 03.500 | 09.000  00.000 00.000 00.000 00.000 | 00.000  ===================  02.000 05.000 01.000 02.000 | 14.000  00.000 05.000 04.000 07.000 | 24.000  00.000 00.000 00.500 03.500 | 09.000  00.000 00.000 00.000 00.000 | 00.000  ===================  4: More than one result found. Assign x4 = 0.  x1 = 22.000, x2 = -09.600, x3 = 18.000, x4 = 00.000  ===================  02.000 \* 22.000 + 05.000 \* -09.600 + 01.000 \* 18.000 + 02.000 \* 00.000 = 14.000  -02.000 \* 22.000 + 00.000 \* -09.600 + 03.000 \* 18.000 + 05.000 \* 00.000 = 10.000  01.000 \* 22.000 + 00.000 \* -09.600 +-01.000 \* 18.000 + 01.000 \* 00.000 = 04.000  00.000 \* 22.000 + 05.000 \* -09.600 + 04.000 \* 18.000 + 07.000 \* 00.000 = 24.000 |

## Netiesinių lygčių sistemų sprendimas

### Grafinis sprendimas



Grafiškai išsprendus, galima matyti, jog sprendinių yra, ir sprendinys gali būti .

## Broideno metodas

Algoritmas baigia darbą jeigu yra pasiekta reikšmė su tam tikru tikslumu, arba viršytas maksimalus iteracijų skaičius. Tikslumas: 1e-12, maksimalus iteracijų skaičius.: 1000.

|  |  |  |
| --- | --- | --- |
| **Pradinis artinys** | **Iteracijų skaičius** | **Gautas rezultatas** |
| 0, 0 | 1000 | null |
| 2, 1 |  | 2.58941 2.08895 |
| 5, 4 |  | 2.58941 2.08895 |
| -2, 1 |  | 2.58941 2.08895 |

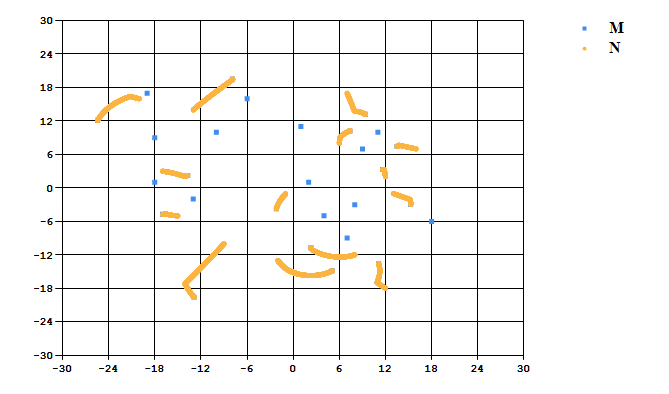
|  |  |  |
| --- | --- | --- |
| **Pradinis artinys** | **Iteracijų skaičius** | **Gautas rezultatas** |
| 0, 0, 0, 0 | 1000 | null |
| 2, 1, 3, 4 |  | -2.44629 0.23848 5.40341 -2.06376 |
| 5, 4, 0 ,1 |  | 1 2 -2 1 |
| -2, 1, -3, -5 |  | 1 2 -2 1 |

## Optimizavimas

Optimizavimo uždavinį spręsti pasirinau greičiausio nusileidimo metodą. Sprendimui reikalinga sudaryti tikslo funkciją. Ši funkcija bus tokia, kad tarp pasirinktų k taškų atstumas būtų kuo artimesnis atstumui tarp tų k taškų vidurkiui.

– vidutinis atstumas tarp visų k taškų.

Ši funkcija yra gana sudėtinga, ir ją reikia minimizuoti. Tam reikalinga išvestinė. Šiuo atveju aš ją aproksimuosiu.

Gautas rezultatas:  


Oranžinės kreivės rodo kiekvieno taško trajektoriją. Ji baigiasi kvadratu.

# Išvados

Tiesinių lygčių sistemą galima itin sparčiai išspręsti taikant Gauso metodą.

Broideno metodas yra efektyvus būdas išspręsti netiesinių lygčių sistemą naudojant išvestinių aproksimaciją.

Naudojant greičiausio nusileidimo metodą galima efektyviai optimizuoti uždavinius.