KAUNO TECHNOLOGIJOS UNIVERSITETAS

Skaitiniai metodai ir algoritmai (P170B115)

Projektas

Trečia užduotis

7 variantas

Atliko:

IFF-8/12 gr. studentas

Jokūbas Akramas

2020 m. lapkričio 30 d.

Priėmė:

lekt. Darius Naujokaitis

KAUNAS 2020

Turinys

1. Interpoliavimas daugianariu.

1.1. Užduotis

1 lentelėje duota interpoliuojamos funkcijos analitinė išraiška. Pateikite interpoliacinės funkcijos išraišką naudodami 1 lentelėje nurodytas bazines funkcijas, kai:

a. Taškai pasiskirstę tolygiai.
b. Taškai apskaičiuojami naudojant Čiobyševo abscises.

Interpoliavimo taškų skaičių parinkite laisvai, bet jis turėtų neviršyti 30. Pateikite du grafikus, kai interpoliacinės funkcijos apskaičiuojamos naudojant skirtingas abscises ir gautas interpoliuojančių funkcijų išraiškas. Tame pačiame grafike vaizduokite duotąją funkciją, interpoliacinę funkciją ir netiktį.

1 lentelė. Interpoliuojamos funkcijos išraiška ir bazinės funkcijos

Var. Nr.	Funkcijos išraiška	Bazinė funkcija
1	$\frac{\ln(x)}{(\sin(2\cdot x)+1{,}5)};2\le x\le 10$	Niutono
2	$\cos(2\cdot x)/(\sin(2\cdot x)+1{,}5)-\cos\frac{x}{5};-2\le x\le 3;$	Čiobyševo
3	$e^{-x^2}\cdot\cos(x^2)\cdot(x+2);\ -2\le x\le 3$	Vienanarių
4	$\frac{\ln(x)}{(\sin(2\cdot x)+1{,}5)}-x/7;2\le x\le 10$	Niutono
5	$\frac{\ln(x)}{(\sin(2\cdot x)+1{,}5)}+\sin\left(\frac{x}{5}\right);2\le x\le 10$	Vienanarių
6	$\frac{\ln(x)}{(\sin(2\cdot x)+1{,}5)}+x/5;2\le x\le 10$	Niutono
7	$\cos(2\cdot x)\cdot(\sin(2\cdot x)+1{,}5)-\cos\frac{x}{5};-2\le x\le 3;$	Čiobyševo

1.2. Teorinė dalis

Uždavinys bus sprendžiamas Čiobyševo metodu, todėl prieš ir po skaičiavimų reikšmes reikės persivesti į Čiobyševo formą ([-1;1]) ir atvirkščiai. Šiuos pervedimus daro metodai *CiobysevoForma()* – eil 16-19 ir *NormaliForma()* - eil 20-23.
Pagal užduoties reikalavimus taškai gali būti pasiskirstę tolygiai arba pagal Čiobyševo abscises. Šis apskaičiavimas vykdomas metode *taskuRinkinys()* – eil 68-89, kur *int pozymis* nusako kuri opcija bus naudojama.
Pagrindinis metodas *Ciobysevas()* – eil 24-67 vykdo skaičiavimus, pradedant nuo daugianarių formavimo cikle, panaudojant metodą *T()* – eil 1-15. Paskui TLS išsprendžiama Gauso metodu, o tada galima išsireikšti daugiklių reikšmes. Apskaičiavus šias reikšmes formuojami vaizdavimo vektoriai, kurie paskui yra panaudojami braižant funkcijos reikšmes.

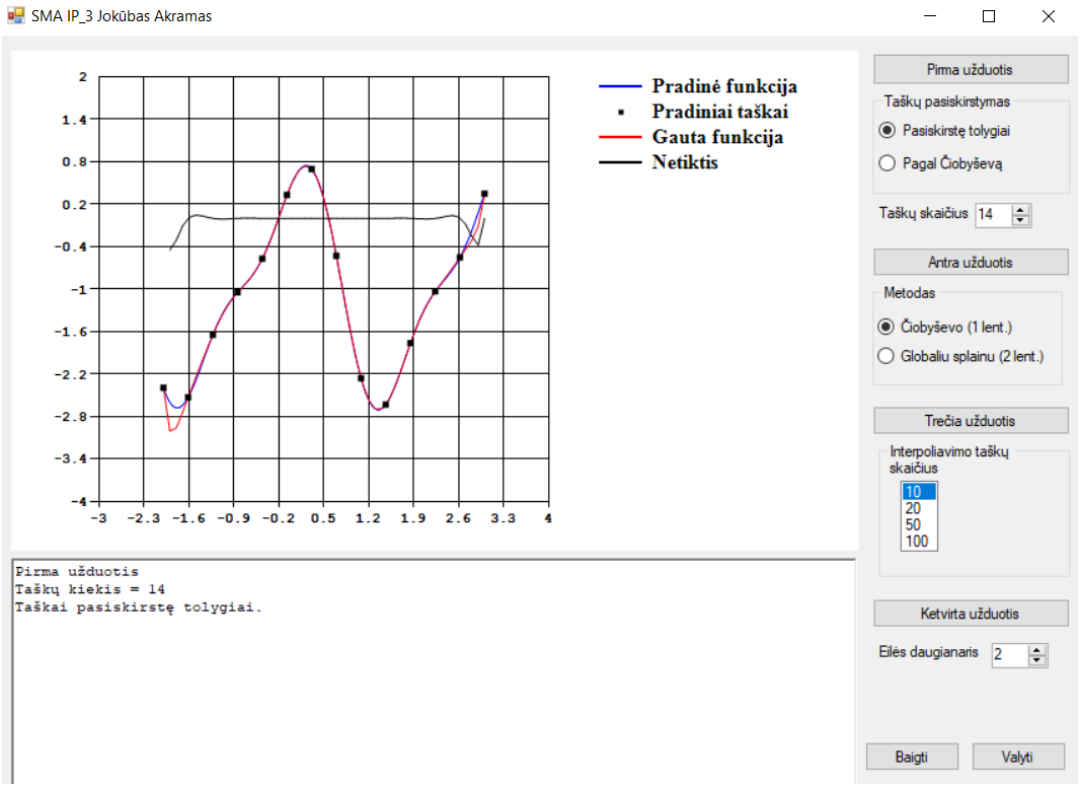
Kodo fragmentas:

```
1.  private double T(double x, int j)
2.  {
3.      if (j == 0)
4.      {
5.          return 1;
6.      }
7.      else if (j == 1)
8.      {
9.          return x;
10.     }
11.     else
12.     {
13.         return 2 * x * T(x, j - 1) - T(x, j - 2);
14.     }
15. }
16. private double CiobysevoForma(double X, double a, double b)
17. {
18.     return ((2 * X) / (b - a)) - ((b + a) / (b - a));
19. }
```

```
20. private double NormaliForma(double X, double a, double b)
21. {
22.     return (((b - a) / 2) * X) + ((b + a) / 2);
23. }
24. private void Ciobysevas(double n, double[] X, double[] taskai, Series z, out double[] X
    Values, out double[] FValues)
25. {
26.     //Čiobyševo daugianarių suvedimas į matricą su y reikšme paskutiniame stulpelyje
27.     double[,] TT = new double[(int)n, (int)n + 1];
28.     for (int i = 0; i < TT.GetLength(0); i++)
29.     {
30.         for (int u = 0; u < TT.GetLength(1) - 1; u++)
31.         {
32.             TT[i, u] = T(CiobysevoForma(taskai[i], X[0], X[1]), u);
33.         }
34.         TT[i, TT.GetLength(1) - 1] = F(taskai[i]);
35.     }
36.     //Čiobyševo daugianarių matricos sprendimas gauso metodu
37.     Gausas(TT, (int)n);
38.     //Išsprendus matricą galime gauti daugiklių reikšmes
39.     double[] AValues = new double[(int)n];
40.     for (int i = 0; i < AValues.Length; i++)
41.     {
42.         AValues[i] = TT[i, (int)n] / TT[i, i];
43.     }
44.     //Interpoliuotos funkcijos reikšmių surašymas į masyvus pagal (x) reikšmes
45.     double deltaX = 0.1;
46.     int N = (int)Math.Round((X[1] - X[0]) / deltaX) + 1;
47.     XValues = new double[N];
48.     FValues = new double[N];
49.     for (int i = 0; i < N; i++)
50.     {
51.         XValues[i] = CiobysevoForma(X[0] + i * deltaX, X[0], X[1]);
52.         FValues[i] = 0;
53.         for (int u = 0; u < (int)n; u++)
54.         {
55.             FValues[i] += T(XValues[i], u) * AValues[u];
56.         }
57.     }
58.     //Gautos interpoliuotos funkcijos braižymas ekrane
59.     if (z != null)
60.     {
61.         for (int i = 0; i < FValues.Length; i++)
62.         {
63.             XValues[i] = NormaliForma(XValues[i], X[0], X[1]);
64.             z.Points.AddXY(XValues[i], FValues[i]);
65.         }
66.     }
67. }
68. private double[] taskuRinkinys(double n, int pozymis, double[] X)
69. {
70.     double[] taskai = new double[(int)n];
71.     if (pozymis == 0)
72.     {
73.         //---Taskai pasiskirste tolygiai
74.         double zingsnis = (X[1] - X[0]) / (n - 1);
75.         for (int i = 0; i < n; i++)
76.         {
77.             taskai[i] = X[0] + zingsnis * i;
78.         }
79.     }
80.     else
81.     {
82.         //---Taskai pasiskirste pagal ciobyseva
83.         for (int i = 0; i < n; i++)
84.         {
85.             taskai[i] = ((X[1] - X[0]) / 2) * Math.Cos(Math.PI * (2 * i + 1) / (2 * n))
    + ((X[1] + X[0]) / 2);
86.         }
87.     }
88.     return taskai;
89. }
```

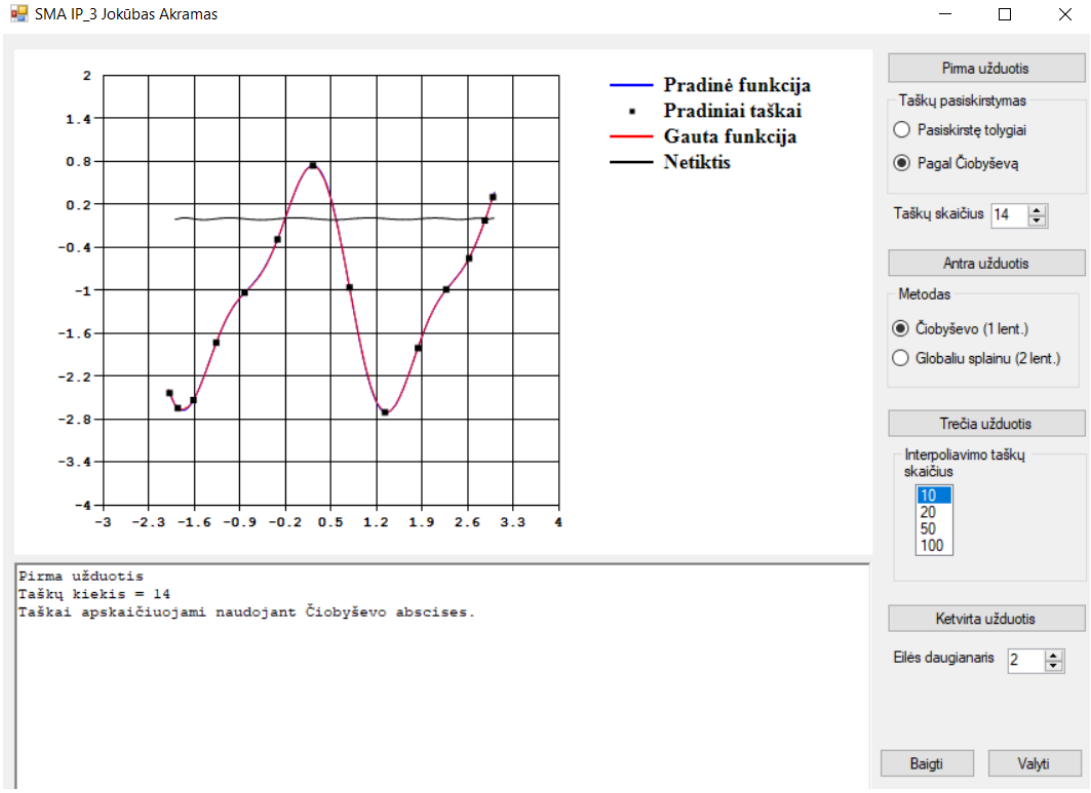
1.3. Rezultatai

Grafike pateikiamas interpoliuojamos funkcijos rezultatas, kai taškai pasiskirstę tolygiai (pav. 1)



pav. 1 Taškai pasiskirstę tolygiai

Kitame grafike pateikiamas interpoliuojamos funkcijos rezultatas, kai taškai pasiskirstę pagal Čiobyševo abscises (pav. 2)



pav. 2 Taškai pasiskirstę pagal Čiobyševo abscises

Iš grafikų puikiai matosi, kad esant tam pačiam interpoliavimo mazgų skaičiui, Čiobyševo abscisių metodas daug geriau paskirsto taškus nei tolyginis paskirstymas – netikties išraiška daug artimesnė 0.

2. Interpoliavimas daugianariu ir splainu per duotus taškus.

2.1. Užduotis

Pagal 2 lentelėje pateiktą šalį ir metus, sudaryti interpoliuojančią kreivę 12 mėnesių temperatūroms atvaizduoti nurodytais metodais:

a. Daugianariu, sudarytu naudojant 1 lentelėje nurodytas bazines funkcijas.
b. 2 lentelėje nurodyto tipo splainu.

2 lentelė. Šalys, metai ir splaino tipas II, III ir IV užduotims.

* pakanka pavaizduoti pagrindinį šalies kontūrą, t. y. nereikia vaizduoti atsiskyrusių teritorijų, pavyzdžiui, šaliai priklausančių salų ir pan.

Var. Nr.	Šalis	Metai	Splainas
1	Rumunija	2008	Globalus
2	Olandija	2010	Ermito (Akima)
3	Vokietija	2003	Globalus
4	Latvija	2015	Ermito (Akima)
5	Kroatija	2017	Globalus
6	Malis	2006	Ermito (Akima)
7	Peru	2008	Globalus

2.2. Teorinė dalis

Kadangi uždavinį reikia spręsti Čiobyševo bazine funkcija arba Globaliu splainu, o Čiobyševo metodą jau aptariau (žr. 1.2 skyrių), todėl šiame skyriuje nagrinėsiu tik globalaus splaino metodą.
Sprendžiant Globalaus splaino metodu reikia žinoti tiek buvusio, tiek sekančio taško koordinates, todėl formuojant matricą sprendimui (eil. 09-16) reikia jas įtraukti su atitinkamais skaičiavimais (eil. 12-14). Taip pat skaičiuojamas reikšmių vektorius (eil. 15). Gauta sprendimo matrica yra [n-2 x n] dydžio, nes kraštiniai taškai nenagrinėjami, dėl to, kad reikalingas i-1 ir i+1 taškas i-tojo taško apskaičiavimui. Vienas iš būdų, kaip galima būtų spręsti, tai laikyti pirmojo ir paskutiniojo taško antrąsias išvestines lygias 0 ir taip paversti sprendimų matricą iš [n-2 x n] į [n-2 x n-2]. Nauja matrica sprendžiama Gauso metodu. Išsprendus TLS iš reikšmių vektoriaus galima išsireikšti antros eilės išvestines kiekvienam taškui (eil. 33-36). Apskaičiavus šias reikšmes formuojamas vaizdavimo vektorius įstatant reikšmes į globalaus splaino išraišką (eil. 51). Gautais vaizdavimo vektoriais braižoma gautos interpoliuotos funkcijos išraiška.

Kodo fragmentas:

```
1.  private void GlobalusSplainas(double[] x, double[] y, Series z)
2.  {
3.      int n = x.Length;
4.      double[] d = new double[n - 1];
5.      for (int i = 0; i < d.Length; i++) d[i] = x[i + 1] - x[i];
6.      double[,] T = new double[n - 2, n];
7.      double[] YY = new double[n - 2];
8.      //Splaino daugianarių matricos [n-2 x n] sudarymas
9.      for (int i = 0; i < T.GetLength(0); i++)
10.     {
11.         for (int u = 0; u < T.GetLength(1); u++) T[i, u] = 0;
12.         T[i, i] = d[i] / 6;
13.         T[i, i + 1] = (d[i] + d[i + 1]) / 3;
14.         T[i, i + 2] = d[i + 1] / 6;
15.         YY[i] = ((y[i + 2] - y[i + 1]) / d[i + 1]) - ((y[i + 1] - y[i]) / d[i]);
16.     }
17.     double[,] TT = new double[n - 2, n - 1];
```

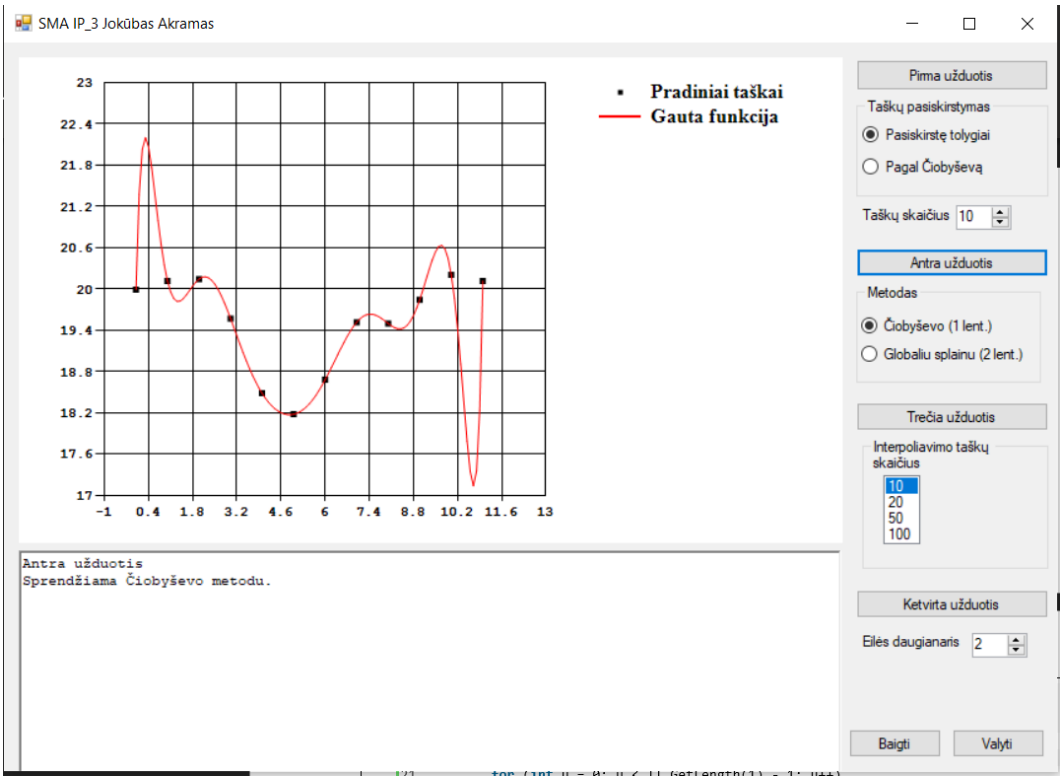
```
    //Splaino daugianarių matricos be 1 ir n-1 reiksmes stulpeliu pervedimas gauso metodu spręsti ir rezultatais paskutiniame stulpelyje
    for (int i = 0; i < TT.GetLength(0); i++)
    {
        for (int u = 0; u < TT.GetLength(1) - 1; u++)
        {
            TT[i, u] = T[i, u + 1];
        }
        TT[i, n - 2] = YY[i];
    }
    //Splaino daugianarių TLS sprendžiama gauso metodu
    Gausas(TT, n - 2);
    double[] f_2 = new double[n];
    f_2[0] = 0;
    f_2[n - 1] = 0;
    //Apskaičiuojamos antros eilės išvestinės iš apskaičiuotos matricos ir y reikšmių vektoriaus
    for (int i = 0; i < n - 2; i++)
    {
        f_2[i + 1] = TT[i, n - 2] / T[i, i];
    }
    for (int i = 0; i < n - 1; i++)
    {
        double xmax = x[i + 1];
        double xmin = x[i];
        double deltaX = 0.01;
        int N = (int)Math.Abs(Math.Round((xmax - xmin) / deltaX)) + 1;

        double[] XValues = new double[N];
        double[] FValues = new double[N];
        //Pagal apskaičiuotas antros eilės išvestines formuojamos vaizdavimo taškų reikšmės XValues ir FValues vektoriuose
        for (int u = 0; u < N; u++)
        {
            XValues[u] = xmin + u * deltaX;
            double s = XValues[u] - x[i];
            FValues[u] = (f_2[i] * (Math.Pow(s, 2) / 2)) - (f_2[i] * (Math.Pow(s, 3) / (6 * d[i]))) + (f_2[i + 1] * (Math.Pow(s, 3) / (6 * d[i]))) + (((y[i + 1] - y[i]) / d[i]) * s) - (f_2[i] * (d[i] / 3) * s) - (f_2[i + 1] * (d[i] / 6) * s) + y[i];
        }

        if (z != null)
        {
            //Gautos interpoliuotos funkcijos braižymas ekrane
            for (int u = 0; u < FValues.Length; u++)
            {
                z.Points.AddXY(XValues[u], FValues[u]);
            }
        }
    }
}
```

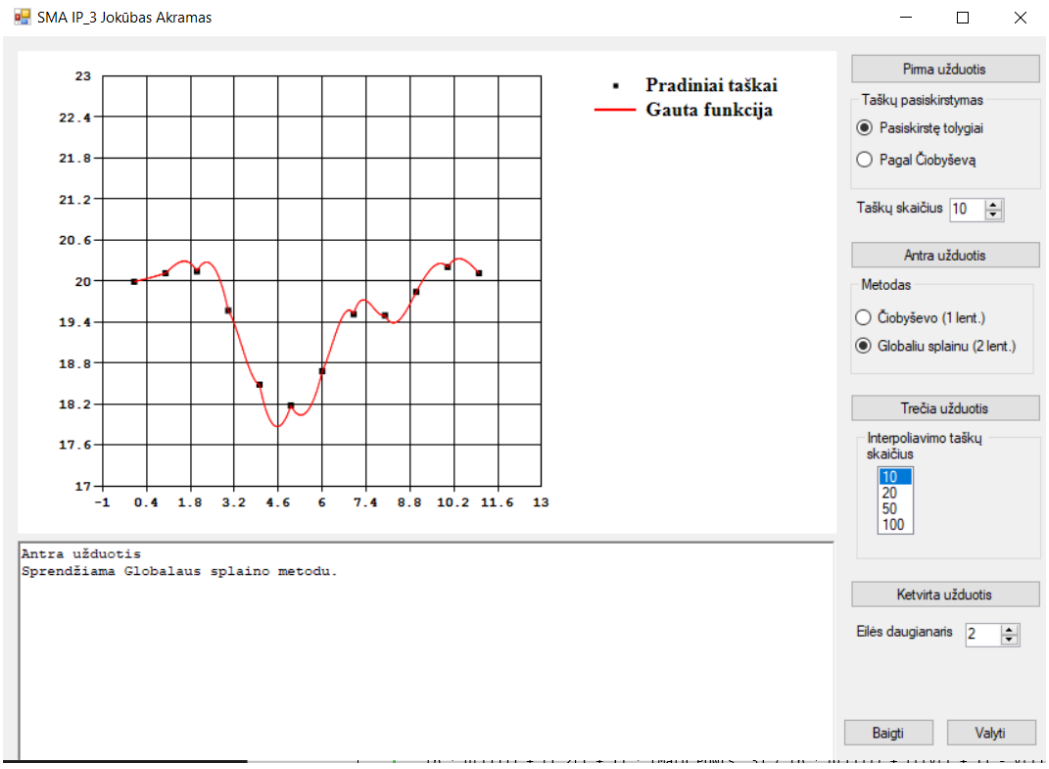
2.3. Rezultatai

Grafike pateikiamas interpoliuojamos funkcijos rezultatas, kai sprendžiama Čiobyševo metodu (pav. 3)



pav. 3 Čiobyševo metodas

Kitame grafike pateikiamas interpoliuojamos funkcijos rezultatas, kai sprendžiama Globalaus splaino metodu (pav. 4)



pav. 4 Globalaus splaino metodas

Iš grafikų aiškiai matyti, kad Čiobyševo metodu gauta funkcija daug labiau linkus „banguoti“, nei Globalaus splaino. Todėl pastaroji dėl šios priežasties yra pranašesnė.

3. Parametrinis interpoliavimas

3.1. Užduotis

Naudodami parametrinio interpoliavimo metodą 2 lentelėje nurodytu splainu suformuokite 2 lentelėje nurodytos šalies kontūrą. Pateikite pradinius duomenis ir rezultatus, gautus naudojant 10, 20, 50, 100 interpoliavimo taškų.

3.2. Teorinė dalis

Kadangi šį uždavinį reikia spręsti Globalaus splaino metodu, o jį jau esu aptaręs ankščiau (žr. 2.2 skyrių), todėl nesikartosiu ir paminėsiu tik esminius principus.
Šiame uždavinyje sprendžiamas parametrinis interpoliavimas, jam pritaikomas Globalaus splaino metodas, tačiau jei ankščiau y koordinatė priklausė nuo x, dabar tiek x tiek y priklausys nuo trečio argumento – t. Todėl pirma globaliu splainu išsprendžiamas x(t) – eil.72, o paskui y(t) – eil. 73. Pagal gautas išraiškas vaizduojama funkcija – eil. 74-81.
Tačiau pirma reikia gauti argumento t reikšmes. Taškai imami iš duoto Peru kontūro koordinačių failų – eil. 31,32. Taškų kiekis apibrėžiamas vartotojo: {10, 20, 50, 100}. Taškai imami kas tam tikrą kiekį elementų, kad pasiskirstymas būtų tolygus – šis veiksmas atliekamas metode *taskai()* – eil. 1-11. Toliau argumentas – vektorius t išreiškiamas atstumu tarp gretimų taškų ilgio šaknimi (Šaknis pradėta naudoti dėl funkcijos eksponentinio bangavimo, kai atstumas tarp taškų yra šiek tiek ilgesnis, nei lyginant su kitais atstumais tarp taškų) – eil. 44, 21.

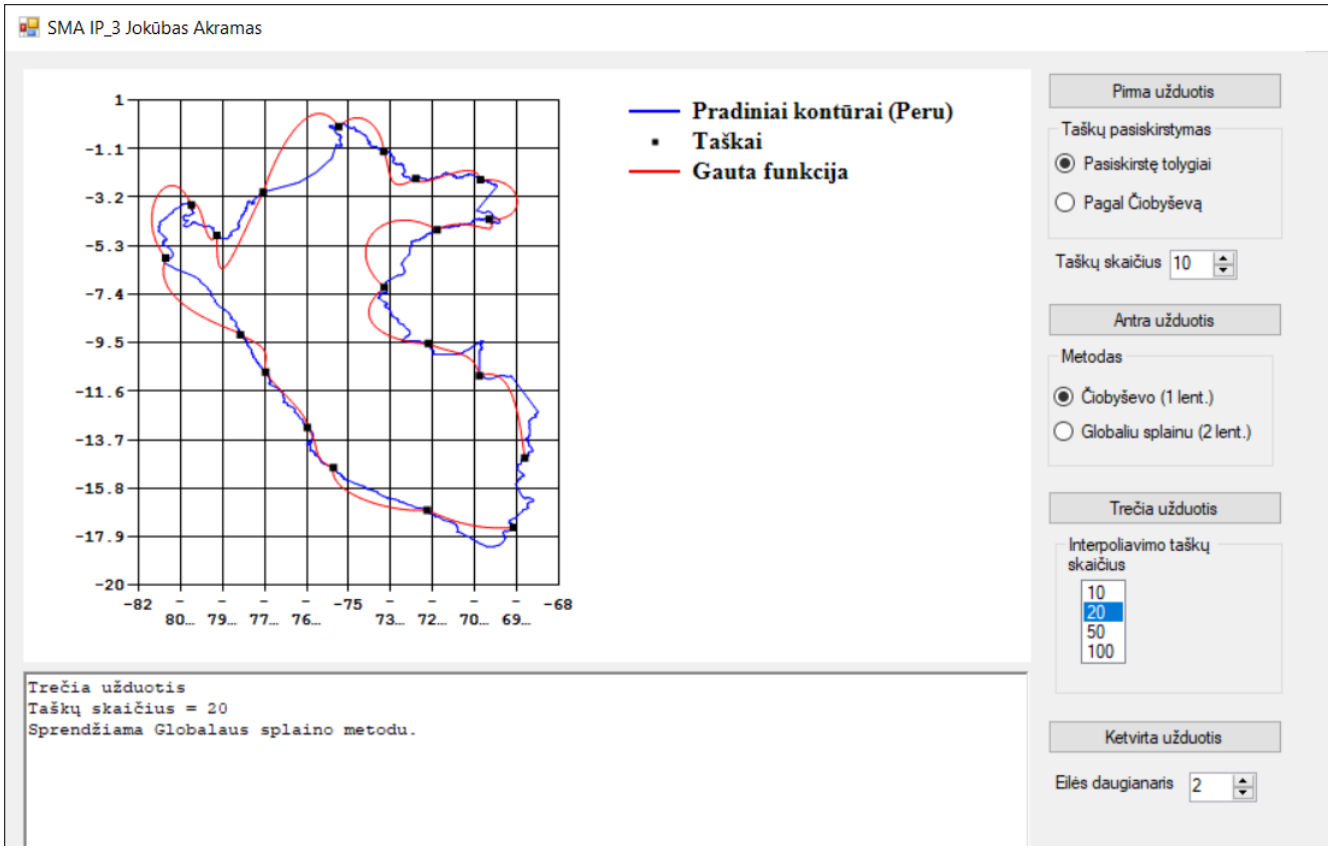
Kodo fragmentas:

```
1.  private double[] taskai(double[] data, int n)
2.  {
3.      double delta = data.Length / n;
4.      double[] tsk = new double[n];
5.      for (int i = 0; i < tsk.Length; i++)
6.      {
7.          int index = (int)Math.Round(i * delta);
8.          tsk[i] = data[index];
9.      }
10.     return tsk;
11. }
12. private double S(double x0, double x1, double y0, double y1)
13. {
14.     //Grąžinama reikšmė t[i] yra taškų [i] ir [i+1] atstumo skirtumo šaknis.
15.     //Šaknis reikalinga tam, kad esant didesniam atstumui tarp vaizdavimo
16.     //taškų funkcija kuo mažiau diverguotų.
17.
18.     //Be šaknies - greitai diverguoja
19.     //return Math.Sqrt(Math.Pow((x1 - x0), 2) + Math.Pow((y1 - y0), 2));
20.
21.     //Su šaknim - lėčiau diverguoja
22.     return Math.Sqrt(Math.Sqrt(Math.Pow((x1 - x0), 2) + Math.Pow((y1 - y0), 2)));
23. }
24. private void button5_Click(object sender, EventArgs e)
25. {
26.     ClearForm1();
27.     PrepareareForm(-82, -68, -20, 1);
28.     //---
29.     int taskuSkaicius = int.Parse((string)listBox1.SelectedItem);
30.     //---
31.     double[] taskaiXData = File.ReadAllLines(@"Data/X.txt")[0].Split(',').Select(i =
    > Double.Parse(i)).ToArray();
32.     double[] taskaiYData = File.ReadAllLines(@"Data/Y.txt")[0].Split(',').Select(i =
    > Double.Parse(i)).ToArray();
33.     double[] taskaiX = taskai(taskaiXData, taskuSkaicius);
34.     double[] taskaiY = taskai(taskaiYData, taskuSkaicius);
35.     double[] taskaiT = new double[taskaiX.Length];
36.     //---
37.     richTextBox1.AppendText("Trečia užduotis\n");
38.     richTextBox1.AppendText("Taškų skaičius = " + taskuSkaicius + "\n");
39.     richTextBox1.AppendText("Sprendžiama Globalaus splaino metodu.\n");
40.     //---
```

```
    taskaiT[0] = 0;
    for (int i = 1; i < taskaiT.Length; i++)
    {
        taskaiT[i] = taskaiT[i - 1] + S(taskaiX[i - 1], taskaiX[i], taskaiY[i - 1], taskaiY[i]);
    }
    //---
    z1 = chart1.Series.Add("Pradiniai kontūrai (Peru)");
    z1.ChartType = SeriesChartType.Line;
    z1.Color = Color.Blue;
    //---
    p1 = chart1.Series.Add("Taškai");
    p1.ChartType = SeriesChartType.Point;
    p1.Color = Color.Black;
    //---
    z2 = chart1.Series.Add("Gauta funkcija");
    z2.ChartType = SeriesChartType.Line;
    z2.Color = Color.Red;
    //---
    for (int i = 0; i < taskaiXData.Length; i++)
    {
        z1.Points.AddXY(taskaiXData[i], taskaiYData[i]);
    }
    for (int i = 0; i < taskaiX.Length; i++)
    {
        p1.Points.AddXY(taskaiX[i], taskaiY[i]);
    }
    //---
    z1.BorderWidth = 1;
    p1.BorderWidth = 3;
    z2.BorderWidth = 1;
    //---
    var xArray = GlobalusSplainasPeru(taskaiT, taskaiX, z2);
    var yArray = GlobalusSplainasPeru(taskaiT, taskaiY, z2);
    for (int i = 0; i < xArray.Length; i++)
    {
        //Gautos interpoliuotos funkcijos braižymas ekrane
        for (int u = 0; u < xArray[i].Length; u++)
        {
            z2.Points.AddXY(xArray[i][u], yArray[i][u]);
        }
    }
}
```

3.3. Rezultatai

Grafike pateikiamas interpoliuojamos parametrinės funkcijos rezultatas, kai sprendžiama Globalaus splaino metodu (pav. 5)



pav. 5 Peru interpoliavimas

4. Aproksimavimas

4.1. Užduotis

Pagal 2 lentelėje nurodytą šalį ir metus mažiausių kvadratų metodu sudarykite aproksimuojančią kreivę 12 mėnesių vidutinėms temperatūroms atvaizduoti naudojant antros, trečios, ketvirtos ir penktos eilės daugianarius. Pateikite gautas daugianarių išraiškas.

4.2. Teorinė dalis

Sprendžiant šį uždavinį reikės išspręsti lygtį:

$$G^T * G * c = G^T * y$$

kur:

G – bazinių funkcijų (x^0, x^1, ..., x^m) matrica (m – eilės daugianaris);
G^T – transponuota G matrica;
c – koeficientų vektorius, kurį reikia rasti;
y – reikšmių vektorius (temperatūros).

Visų pirma reikia apskaičiuoti G ir G^T matricas, Y reikšmių vektorių – visa tai atliekama cikle for – eil. 8-16. Paskui abi lygties pusės sudauginamos, išreiškiama viena matrica *A* TLS spręsti. Matrica A išsprendžiama Gauso metodu – eil. 34, ko pasekoje galima išsireikšti koeficientus c – eil. 38-41. Gavus koeficientų vektorių *c* formuojami funkcijos vaizdavimo vektoriai – eil. 48-56. Paskui gauta funkcija braižoma – eil. 58-64 ir išvedama gauta daugianarė funkcija – eil. 66-77.

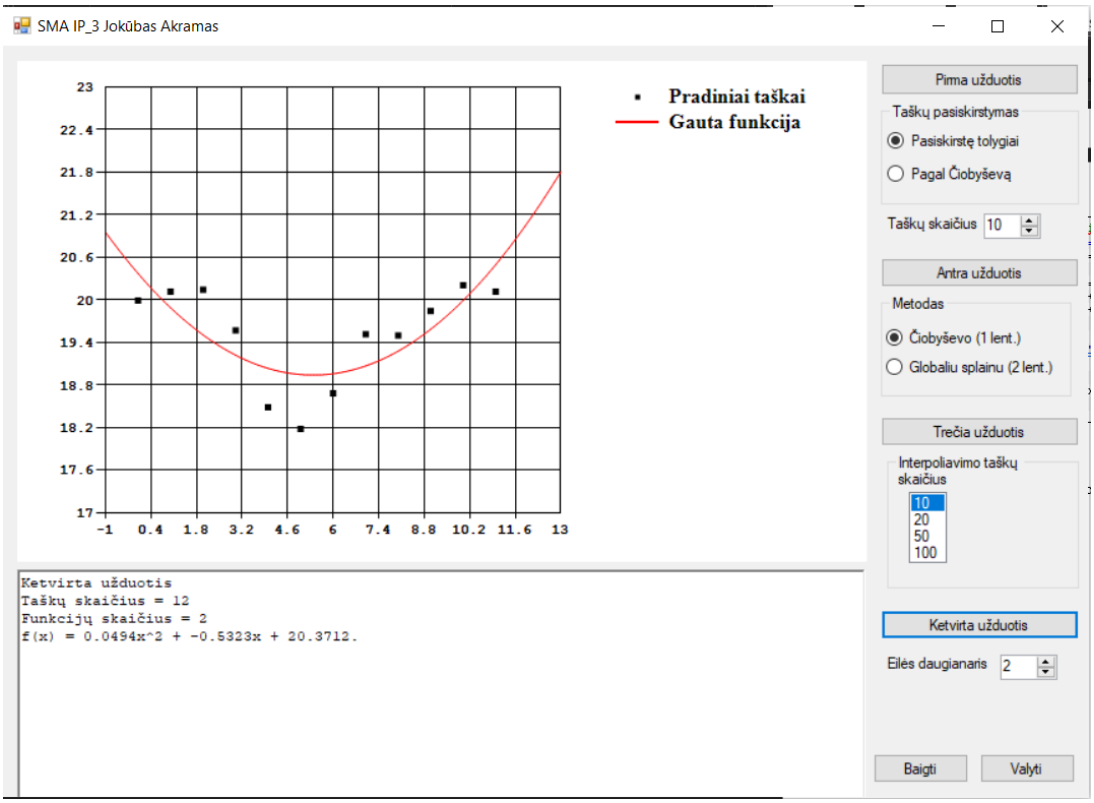
Kodo fragmentas:

```
1.  private void Aproksimavimas(int m, double[] X, double[] x, double[] y, Series z)
2.  {
3.      int n = x.Length;
4.      double[,] G = new double[n, m];
5.      double[,] GT = new double[m, n];
6.      double[,] Y = new double[n, 1];
7.      //G ir G transponuotos matricų gavimas pagal pateiktas x koordinates
8.      for (int i = 0; i < n; i++)
9.      {
10.         for (int u = 0; u < m; u++)
11.         {
12.             G[i, u] = Math.Pow(x[i], u);
13.             GT[u, i] = G[i, u];
14.         }
15.         Y[i, 0] = y[i];
16.     }
17.     //---
18.     //Apskaičiuojamos abi lygties pusės (be C vektoriaus)
19.     double[,] GTG = mulMatrix(GT, G);
20.     double[,] GTY = mulMatrix(GT, Y);
21.     //---
22.     //Iš abiejų lygties pusių formuojama matrica TLS sprendimui
23.     double[,] A = new double[m, m + 1];
24.     for (int i = 0; i < A.GetLength(0); i++)
25.     {
26.         for (int u = 0; u < A.GetLength(1)-1; u++)
27.         {
28.             A[i, u] = GTG[i, u];
29.         }
30.         A[i, A.GetLength(1) - 1] = GTY[i, 0];
31.     }
32.     //---
33.     //TLS sprendimas gauso metodu
34.     Gausas(A, m);
35.     //---
36.     //Apskaičiuojamos C vektoriaus reikšmės
37.     double[] CValues = new double[m];
38.     for (int i = 0; i < CValues.Length; i++)
39.     {
40.         CValues[i] = A[i, m] / A[i, i];
41.     }
42.     //---
43.     double deltaX = 0.1;
44.     int N = (int)Math.Round((X[1] - X[0]) / deltaX) + 1;
45.     double[] XValues = new double[N];
46.     double[] FValues = new double[N];
47.     //Skaičiuojama vaizdavimo matrica (XValues[], FValues[])
48.     for (int i = 0; i < N; i++)
49.     {
50.         XValues[i] = X[0] + i * deltaX;
51.         FValues[i] = 0;
52.         for (int u = 0; u < m; u++)
53.         {
54.             FValues[i] += Math.Pow(XValues[i], u) * CValues[u];
55.         }
56.     }
57.     //Braižoma funkcija pagal vaizdavimo matricą
58.     if (z != null)
59.     {
60.         for (int i = 0; i < FValues.Length; i++)
61.         {
62.             z.Points.AddXY(XValues[i], FValues[i]);
63.         }
64.     }
```

```
65.     //Išvedamos daugianarių išraiškos
66.     string fname = "f(x) = ";
67.     for (int i = m-1; i >= 0; i--)
68.     {
69.         if(i > 1) fname += string.Format("{0, 0:F4}x^{1}", CValues[i], i);
70.         if(i == 1) fname += string.Format("{0, 0:F4}x", CValues[i]);
71.         if(i == 0) fname += string.Format("{0, 0:F4}", CValues[i]);
72.         if (i - 1 != -1)
73.         {
74.             fname += " + ";
75.         }
76.     }
77.     richTextBox1.AppendText(fname + ".\n");
78. }
```

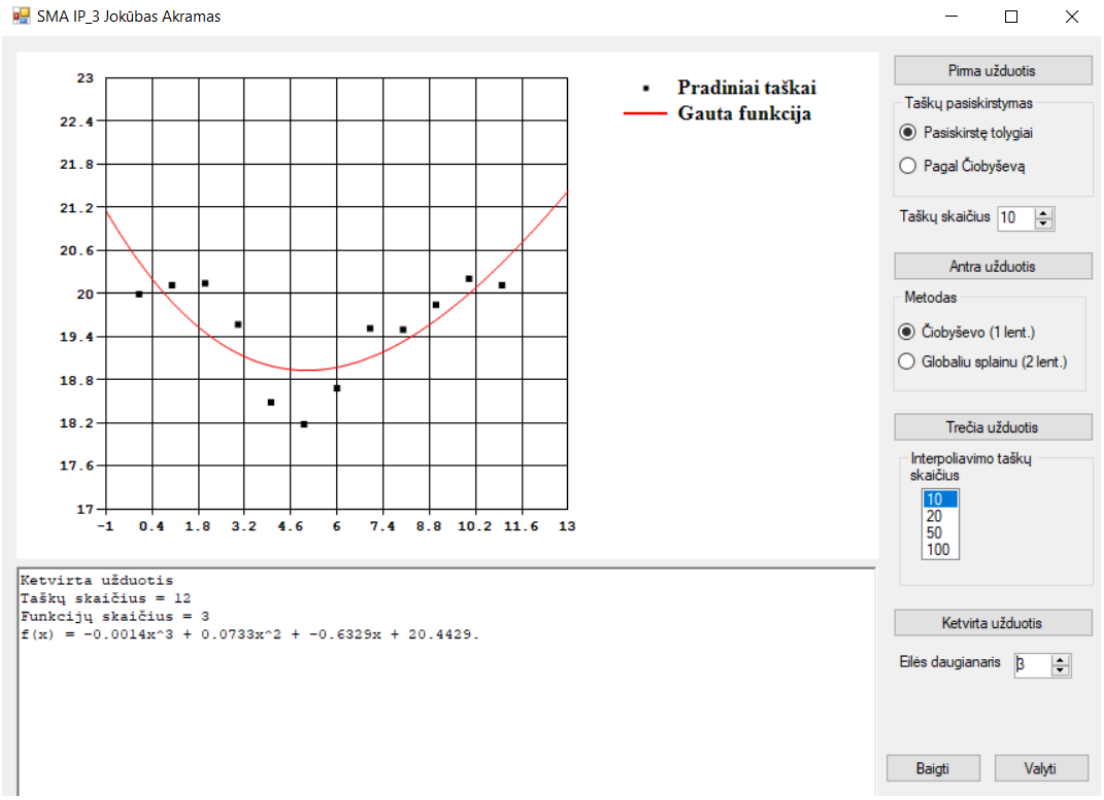
4.3. Rezultatai

Grafike pateikiamas aproksimuojančios kreivės antros eilės daugianaris (pav. 6)



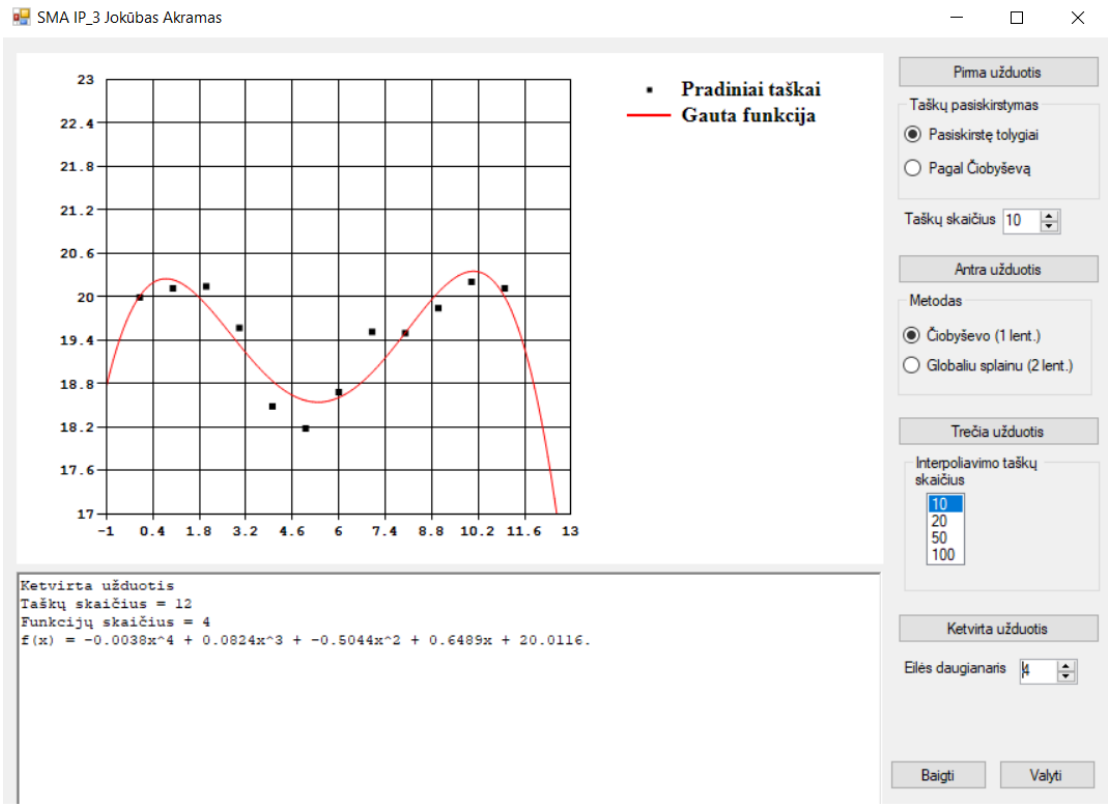
pav. 6 Antros eilės aproksimuojantis daugianaris

Grafike pateikiamas aproksimuojančios kreivės trečios eilės daugianaris (pav. 7)



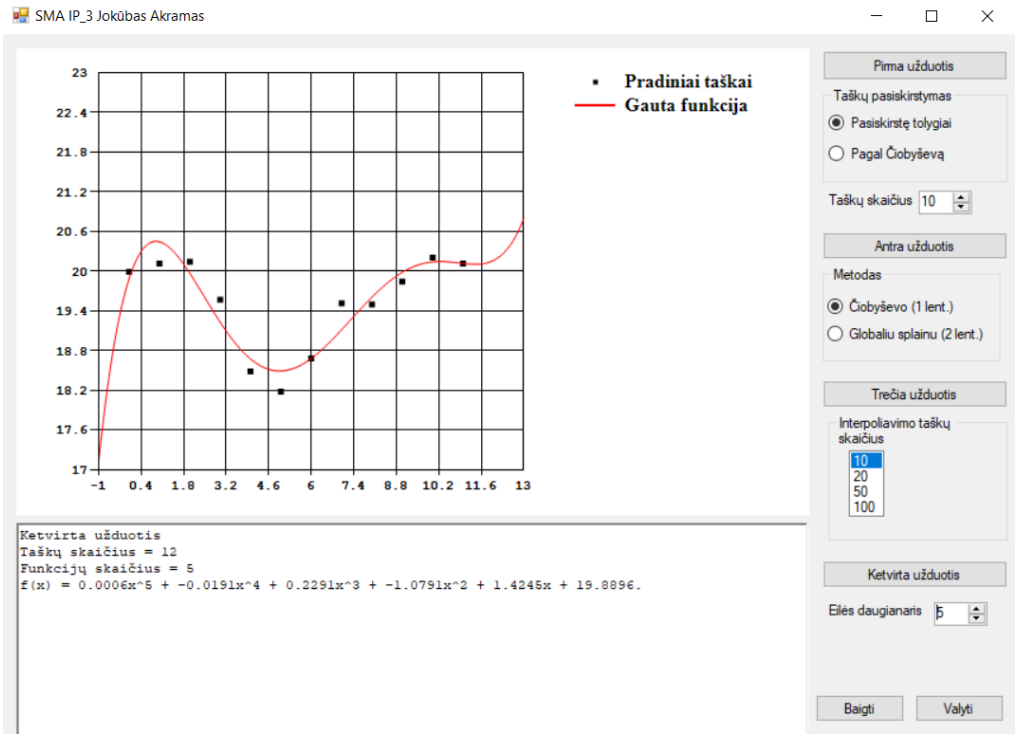
pav. 7 Trečios eilės aproksimuojantis daugianaris

Grafike pateikiamas aproksimuojančios kreivės ketvirtos eilės daugianaris (pav. 8)



pav. 8 Ketvirtos eilės aproksimuojantis daugianaris

Grafike pateikiamas aproksimuojančios kreivės penktos eilės daugianaris (pav. 9)



pav. 9 Penktos eilės aproksimuojantis daugianaris

5. Išvados

Darant šį darbą buvo įsisavinti interpoliavimo metodai.
Išsiaiškinta, kaip Čiobyševo abscisės pagerina rezultatą palyginus su tolygiu taškų pasiskirstymu.
Palygintas Čiobyševo metodo ir Globalaus splaino rezultatas pateikus tuos pačius duomenis.
Įsisavintas parametrinis interpoliavimas Globaliu splainu.
Sudarytas algoritmas įvairių aproksimuojančių eilės daugianarių kreivėms sudaryti remiantis 2008 Peru temperatūrų duomenimis.