KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

TAIKOMOSIOS INFORMATIKOS KATEDRA

SKAITINIAI METODAI IR ALGORITMAI(P170B115)

3 LABORATORINIS DARBAS

Varianto Nr. 5

Atliko:

IFF-1/8 gr. studentas

Matas Palujanskas

Priėmė:

Prof. Rimantas Barauskas

Doc. Andrius Kriščiūnas

KAUNAS

2023

Matas Palujanskas

1. Turinys

Matas Palujanskas

1. Pirma užduoties dalis

1.1 Užduoties sąlygos

I užduotis. Interpoliavimas daugianariu.

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 interpoliuojančios funkcijos apskaičiuojamos naudojant skirtingas abscises ir gautas interpoliuojančių funkcijų išraiškas. Tame pačiame grafike vaizduokite duotąją funkciją, interpoliuojančią funkciją ir netiktį.

pav. 1 Antrojo laboratorinio pirmos dalies užduotys

Užduoties variantas: 5

Var. Nr.	Funkcijos išraiška	Bazinė funkcija
1	$e^{-x^2} \cdot \sin(x^2) \cdot (x-3);\ -3 \le x \le 2$	Čiobyševo
2	$\frac{\ln(x)}{(\sin(2\cdot x)+1{,}5)} + x/5; 2 \le x \le 10$	Vienanarių
3	$e^{-x^2} \cdot \cos(x^2) \cdot (x-3);\ -3 \le x \le 2$	Čiobyševo
4	$\cos(2\cdot x)\cdot(\sin(2\cdot x)+1{,}5)+\cos x; -2 \le x \le 3;$	Vienanarių
5	$e^{-x^2} \cdot \sin(x^2) \cdot (x-3);\ -3 \le x \le 2$	Čiobyševo
6	$\frac{\ln(x)}{(\sin(2\cdot x)+1{,}5)} + x/5; 2 \le x \le 10$	Vienanarių
7	$\cos(2\cdot x)\cdot(\sin(2\cdot x)+1{,}5)-\cos\frac{x}{5}; -2 \le x \le 3;$	Čiobyševo
8	$\frac{\ln(x)}{(\sin(2\cdot x)+1{,}5)} + x/5; 2 \le x \le 10$	Vienanarių
9	$\cos(2\cdot x)\cdot(\sin(2\cdot x)+1{,}5)+\cos x; -2 \le x \le 3;$	Čiobyševo
10	$\cos(2\cdot x)\cdot(\sin(3\cdot x)+1{,}5)-\cos\frac{x}{5}; -2 \le x \le 3;$	Vienanarių

pav. 2 Interpoliuojamų funkcijų išraiškų lentelė

5	$e^{-x^2} \cdot \sin(x^2) \cdot (x-3);\ -3 \le x \le 2$	Čiobyševo

pav. 3 5 varianto funkcijos išraiška

1.2 Interpoliavimas daugianariu

Sprendimas:

Uždavinys bus sprendžiamas Čiobyševo metodu, todėl reikšmes reikės persivesti į Čiobyševo formą ir atvirkščiai. Šiuos pervedimus daro metodai `ciobysevo_mazgas`. Pagal užduoties reikalavimus taškai gali būti pasiskirstę tolygiai arba pagal Čiobyševo abscises. Apskaičiavus šias reikšmes formuojami vaizdavimo vektoriai, kurie paskui yra panaudojami braižant funkcijos reikšmes.

$$x_k = \frac{1}{2} * (a+b) + \frac{1}{2} * (b-a) * (\cos(\frac{2k-1}{2n}) * \pi), k = 1, .., n$$

pav. 4 Čiobyševo abscisės

1.2.1 Sprendimo kodas

Lab3-1.py:

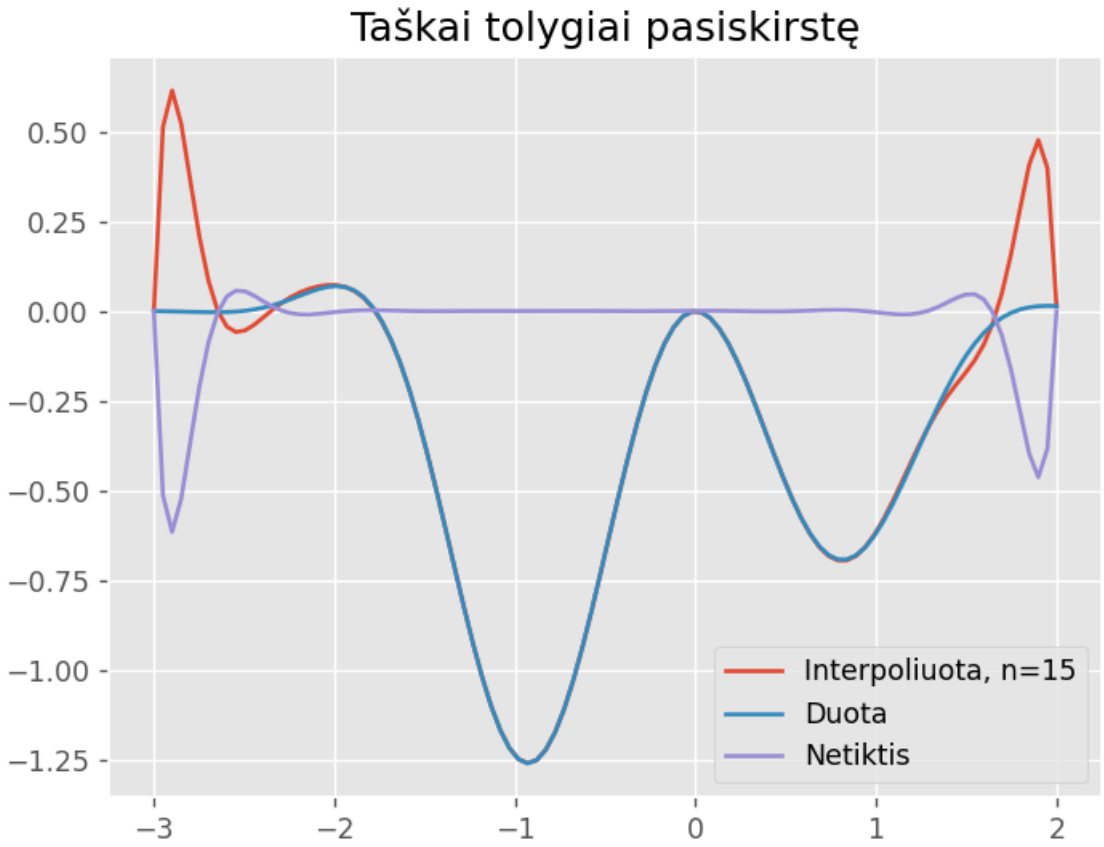
```
import numpy as np
import matplotlib
matplotlib.use('TkAgg')
from numpy import sin, cos, arccos, pi, exp
import matplotlib.pyplot as plt
import sympy as sym

plt.style.use('ggplot')

# Duota funkcija
def funkcija(x):
    return (exp(-x**2)) * sin(x**2) * (x-3)

# Čiobyševo polinomo formulė
def ciobysevo_daugianaris(x, i):
    return cos(arccos(x) * i)

# Čiobyševo polinomo funkcijos formulė spausdinimui
def sym_ciobysevo_daugianaris(x, i):
    return sym.cos(sym.acos(x) * i)

# Čiobyševo intervalo formulė
def ciobysevo_intervalas(x, a, b):
    return (2 * x - (b + a)) / (b - a)

# Čiobyševo mazgo transformacija
def ciobysevo_mazgas(i, pr, pb, n):
    return ((pb - pr) / 2) * cos(pi * (2 * i + 1) / (2 * n)) + ((pb + pr) / 2)
```

Matas Palujanskas

```
# Spausdinimo funkcija
def spausdinti_ciobysevo(i, koffs):
    x = sym.Symbol('x')
    koffs = koffs.flatten()
    xc = ciobysevo_intervalas(x, -3, 2)
    for i in range(len(koffs)):
        A = sym_ciobysevo_daugianaris(xc, i)
        if i == 0:
            print(str(koffs[i]) + ' +')
        elif i == 1:
            print(str(koffs[i]) + ' * (' + str(A) + ') +')
        else:
            print(str(koffs[i]) + ' * ' + str(A) + ' +')

print('Pasirinkite taškų išdėstymo tipą:')
print('Tolygiai - tl')
print('Čiobyševo - cb')

n = 15  # interpoliavimo taškų skaičius
i = np.arange(n)

if str(input("Pasirinkta: ")) == "tl":
    x = np.linspace(-3, 2, n).reshape(-1, 1)
    plot_name = 'Taškai tolygiai pasiskirstę'
else:
    x = ciobysevo_mazgas(i, -3, 2, n).reshape(-1, 1)
    plot_name = ' Taškai naudojant Čiobyševo abscises'

int_ciobysevo = ciobysevo_intervalas(x, -3, 2)
cb_daugianaris = ciobysevo_daugianaris(int_ciobysevo, i)
koffs = np.linalg.solve(cb_daugianaris, funkcija(x))

x = np.linspace(-3, 2, 100).reshape(-1, 1)

cb_i_intervalas = ciobysevo_intervalas(x, -3, 2)
cb_i_daugianaris = ciobysevo_daugianaris(cb_i_intervalas, i)
int = np.dot(cb_i_daugianaris, koffs)

plt.plot(x, int, label='Interpoliuota, n=15')
plt.plot(x, funkcija(x), label='Duota')
plt.plot(x, funkcija(x) - int, label='Netiktis')
plt.legend()
plt.title(plot_name)
plt.show()
```

1.2.2 Gauti rezultatai

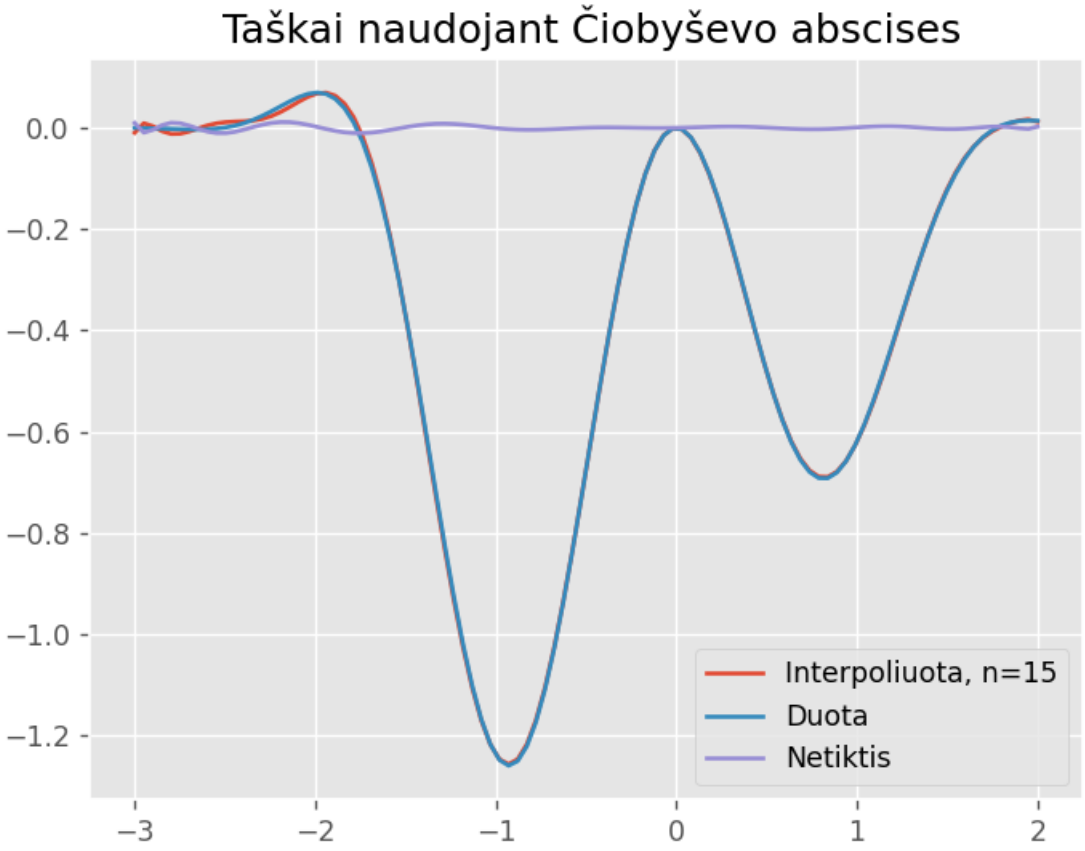
Pirmame grafike atvaizduojamas interpoliuojamos funkcijos rezultatas, kai taškai pasiskirstę tolygiai:



pav. 5 Daugianario interpoliacija tolygiai

Mėlyna spalva pažymėta duota funkcija, violetine netiktis, interpoliuota raudona, buvo naudota 15 taškų.

Antrame grafike vaizduojamas interpoliuojamos funkcijos rezultatas, kai taškai pasiskirstę pagal Čiobyševo abscises:



pav. 6 Daugianario interpoliacija pagal Čiobyševą

Mėlyna spalva pažymėta duota funkcija, violetine netiktis, interpoliuota raudona, buvo naudota taip pat 15 taškų.

Išvada: galima pastebėti, kad netiktis prie Čiobyševo interpoliacijos yra sumažinta kraštiniuose intervalo taškuose. Tai būtent yra Čiobyševo abscisės tikslas - optimaliai pasiskirstyti taškus taip, kad sumažėtų interpoliacijos netiktis kraštiniuose intervalo taškuose. Taigi, galima patvirtinti, kad Čiobyševo metodas taškus paskirsto geriau nei tolyginis paskirstymas.

Matas Palujanskas

2. Antra užduoties dalis

2.1. Užduoties sąlygos

II užduotis. Interpoliavimas splainu per duotus taškus

Sudarykite ***2 lentelėje*** nurodytos šalies 1998-2018 metų šiltnamio dujų emisiją (galimo duomenų šaltinio nuoroda apačioje) interpoliuojančias kreives, kai interpoliuojama ***2 lentelėje*** nurodyto tipo splainu. Pateikite rezultatų grafiką (interpoliavimo mazgus ir gautą kreivę (vaizdavimo taškų privalo būti daugiau nei interpoliavimo mazgų)).

pav. 7 Antros dalies sąlygos

5 varianto šalis:

5	Kroatija	Globalus

pav. 8 5 varianto šalis ir splaino metodas

$$f''_{i-1}\frac{d_{i-1}}{6}+f''_{i}\frac{d_{i-1}+d_i}{3}+f''_{i+1}\frac{d_i}{6}=\frac{y_{i+1}-y_i}{d_i}-\frac{y_i-y_{i-1}}{d_{i-1}}$$

pav. 9 Splaino koeficientai

$$\begin{bmatrix} \frac{d_1}{6} & \frac{d_1+d_2}{3} & \frac{d_2}{6} & 0 & \cdots & 0 \\ 0 & \frac{d_2}{6} & \frac{d_2+d_3}{3} & \frac{d_3}{6} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots\cdots & \vdots \\ 0 & 0 & \cdots & \frac{d_{n-2}}{6} & \frac{d_{n-2}+d_{n-1}}{3} & \frac{d_{n-1}}{6} \end{bmatrix} \begin{Bmatrix} f''_1 \\ f''_2 \\ \vdots \\ f''_n \end{Bmatrix} = \begin{Bmatrix} \frac{y_3-y_2}{d_2}-\frac{y_2-y_1}{d_1} \\ \frac{y_4-y_3}{d_3}-\frac{y_3-y_2}{d_2} \\ \vdots \\ \frac{y_n-y_{n-1}}{d_{n-1}}-\frac{y_{n-1}-y_{n-2}}{d_{n-2}} \end{Bmatrix}$$

pav. 10 Antrų išvestinių skaičiavimų matricos

Apskaičiavus antrų išvestinių reikšmes galima skaičiuoti splainus:

$$_{(i)}f(s)=f''_i\frac{s^2}{2}-f''_i\frac{s^3}{6d_i}+f''_{i+1}\frac{s^3}{6d_i}+\frac{y_{i+1}-y_i}{d_i}s-f''_i\frac{d_i}{3}s-f''_{i+1}\frac{d_i}{6}s+y_i;$$

pav. 11 Antrų išvestinių reikšmių gavimas

Matas Palujanskas

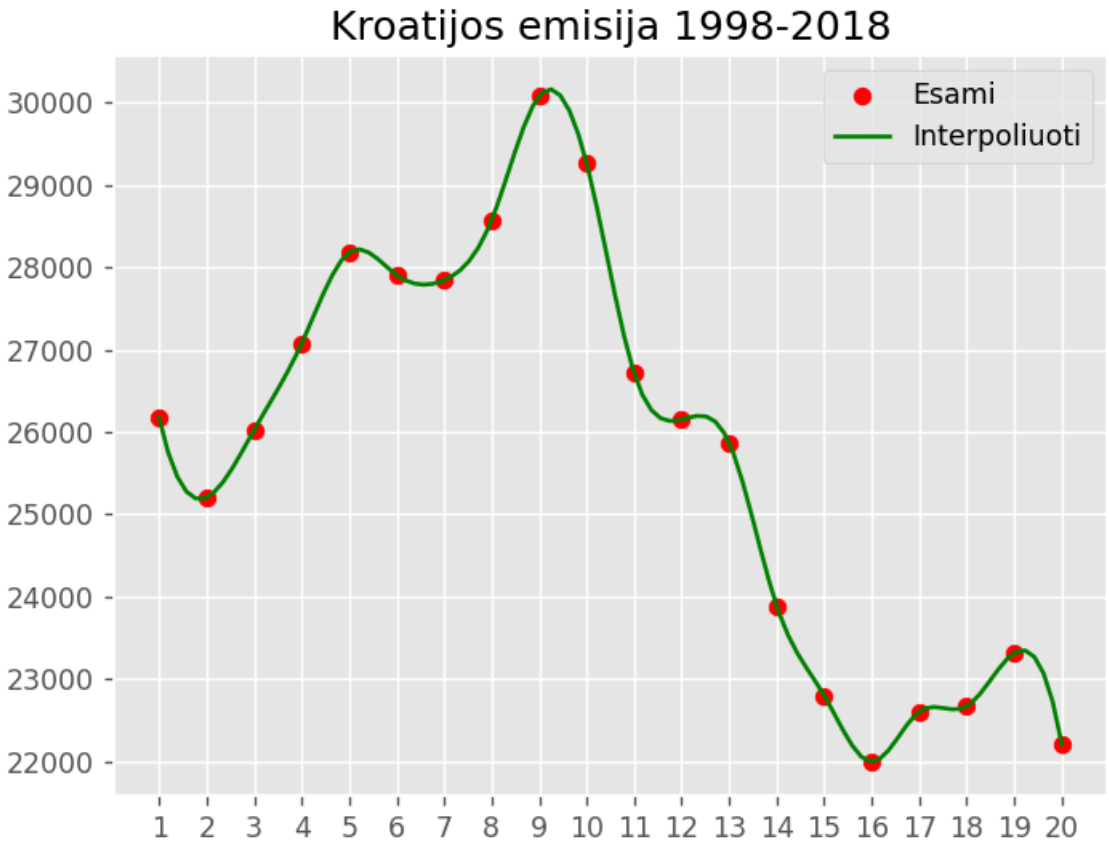
2.2. Sprendimo kodas

Lab3-2.py:

```
import numpy as np
import matplotlib
matplotlib.use('TkAgg')
import matplotlib.pyplot as plt
from scipy.interpolate import UnivariateSpline
import pandas as pd

plt.style.use('ggplot')

def interpoliuoti_globalu(x, y):
    # Surandame interpoliacinį splainą
    spline = UnivariateSpline(x.ravel(), y.ravel(), s=0)

    # Rekonstruojame reikšmes naujuose taškuose
    x_n = np.linspace(x.min(), x.max(), 100)
    y_n = spline(x_n)

    plt.title('Kroatijos emisija 1998-2018 (Globalus)')
    plt.scatter(x, y, label='Esami', color='red')  # Scatter plot for existing data
points
    plt.plot(x_n, y_n, 'g-', label='Interpoliuoti')
    plt.legend()
    plt.xticks(np.arange(20), np.arange(1, 21))
    plt.show()

n = 20  # 20 metų
data = pd.read_csv('Croatia_Emissions.csv')
x = data.index.to_numpy().reshape(-1, 1)
y = data.iloc[:, 0].to_numpy().reshape(-1, 1)
del data

interpoliuoti_globalu(x, y)
```

2.3. Gauti rezultatai

Atvaizduota interpoliuojanti kreivė Globaliu splaino tipu:



pav. 12 Kroatijos emisija

Išvada: globalaus metodo splainas yra pakankamai tikslus, taip pat buvo patogu naudoti `scipy.interpolate` biblioteką.

Matas Palujanskas

3. Trečia dalis

3.1 Užduoties sąlyga

III užduotis. Aproksimavimas

Mažiausių kvadratų metodu sudarykite ***2 lentelėje*** nurodytos šalies 1998-2018 metų šiltnamio dujų emisiją (galimo duomenų šaltinio nuoroda apačioje) aproksimuojančias kreives (**pirmos, antros, trečios** ir **penktos** eilės daugianarius). Pateikite gautas daugianarių išraiškas ir grafinius rezultatus.

pav. 13 Trečios dalies sąlyga

3.2 Sprendimo kodas

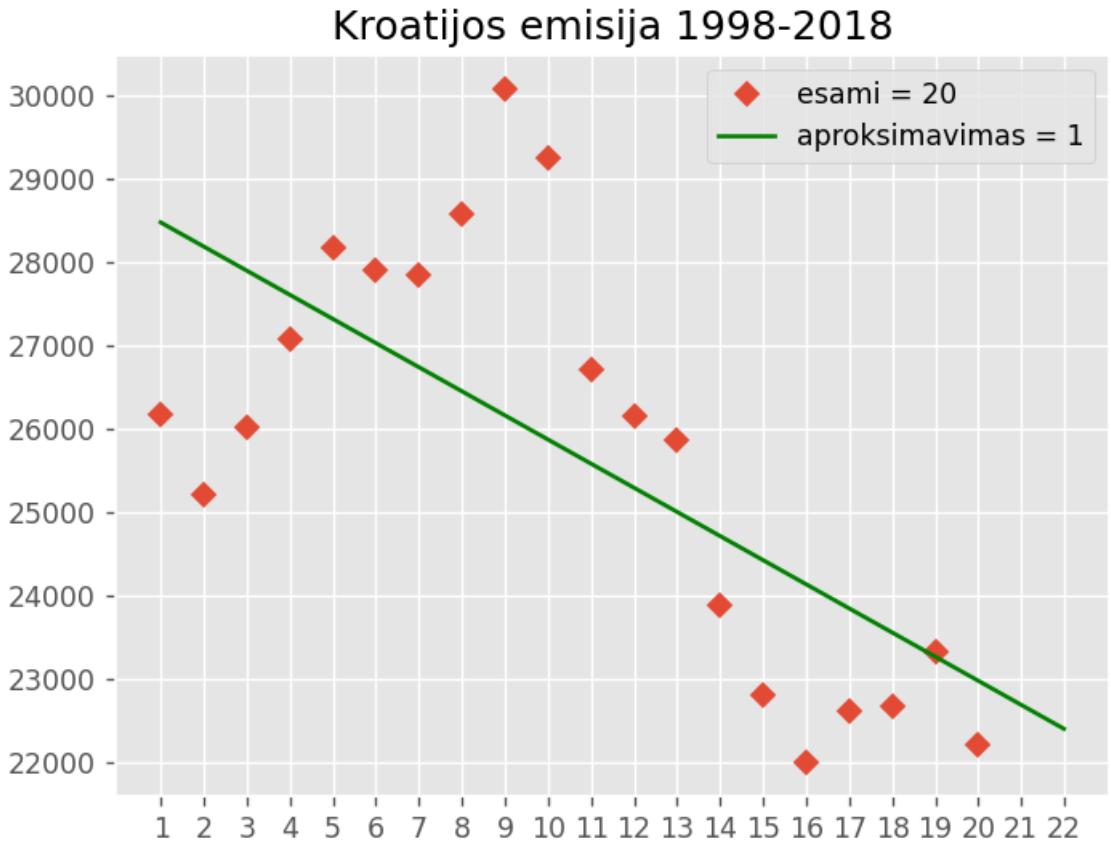
Lab3-3.py:

```
import numpy as np
import matplotlib
matplotlib.use('TkAgg')
import pandas as pd
import matplotlib.pyplot as plt

plt.style.use('ggplot')


def kelti(x, laipsnis):
    x = x.flatten()  # iš 2d į 1d
    return np.array([x ** i for i in range(laipsnis)]).T.astype(float)  # skaičius
pakeliamas tam tikru laipsniu

def b(x, koffs):  # aproksimavimui x pakelti ir su koffu sudaugint
    koffs = koffs.flatten()  # iš 2d į 1d
    ats = 0
    for i in range(len(koffs)):
        ats += koffs[i] * x ** i
        print(f'{koffs[i]} * x ^ {i} +')
    return ats

laipsnis = int(input('Daugianario laipsnis: '))
laipsnis += 1
duomenys = pd.read_csv('Croatia_Emissions.csv')
x = duomenys.index.to_numpy().reshape(-1, 1)  # indeksai/mėnesiai
y = duomenys.iloc[:, 0].to_numpy().reshape(-1, 1)  # temperatūros

G = kelti(x, laipsnis)
koffs = np.linalg.solve(G.T.dot(G), G.T.dot(y))  # išsprendžia lygčių sistema

xx = np.arange(0, 21, 0.01).reshape(-1, 1)  # į 2d

plt.plot(x, y, 'D', label=f'esami = {len(x)}')
plt.plot(xx, b(xx, koffs), 'g-', label=f'aproksimavimas = {laipsnis-1}')
plt.xticks(np.arange(22), np.arange(1, 23))
plt.title('Kroatijos emisija 1998-2018')
```

```
plt.legend()
plt.show()
```

3.3 Gauti rezultatai

3.3.1 Pirmos eilės daugianaris

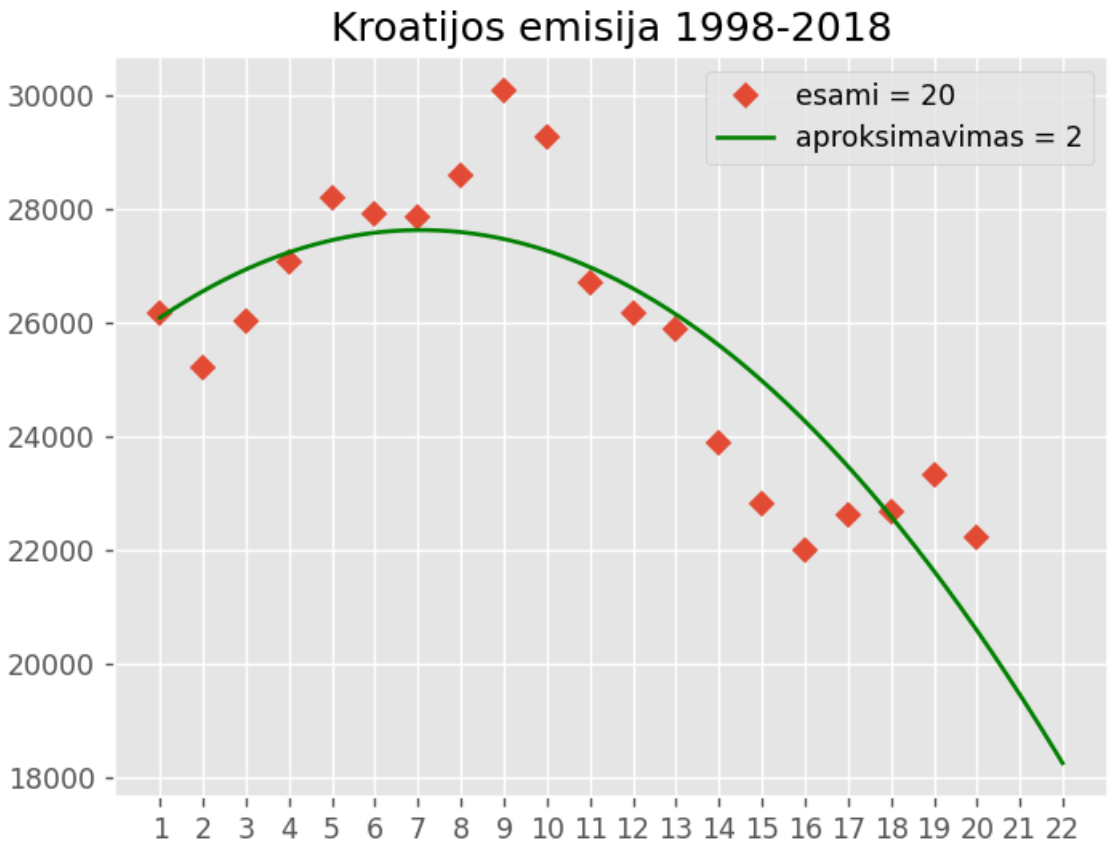


pav. 14 Pirmos eilės daugianario grafinis atvaizdavimas

```
Daugianario laipsnis: 1
28477.01995 * x ^ 0 +
-289.4678127368422 * x ^ 1 +
```

pav. 15 Pirmos eilės daugianario išraiška

3.3.2 Antros eilės daugianaris

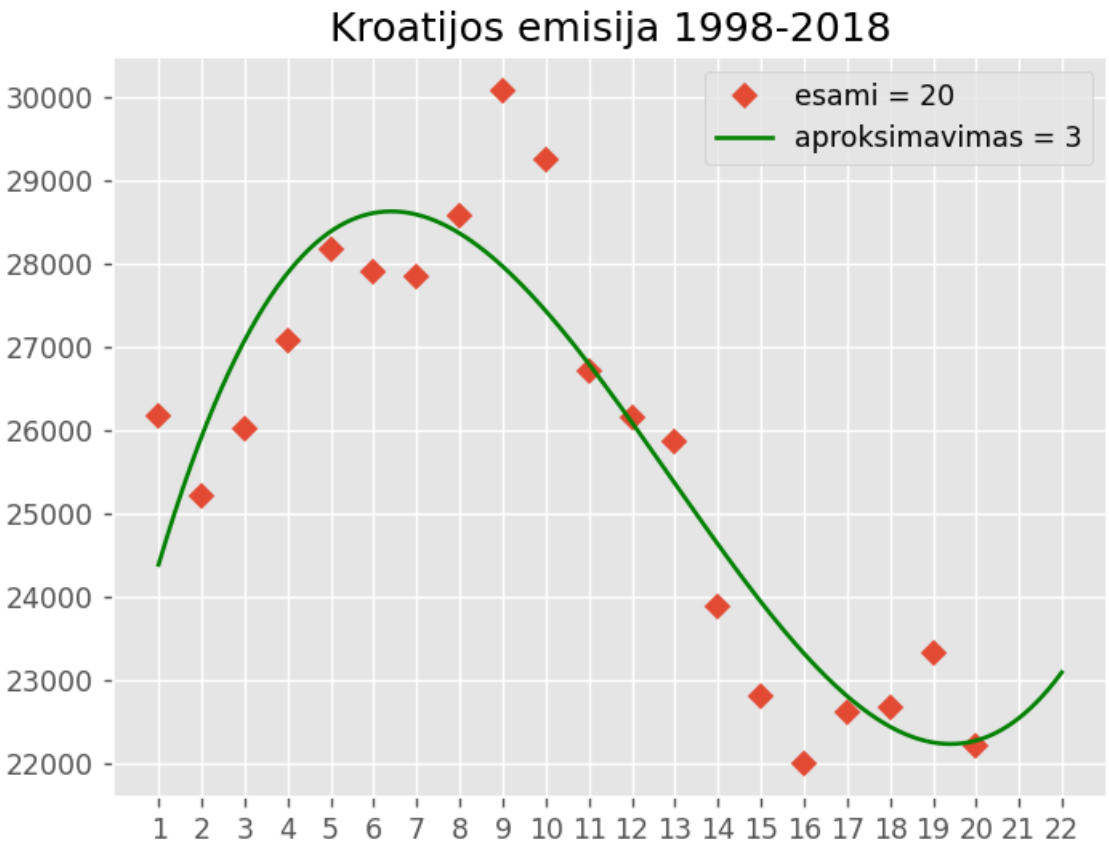


pav. 16 Antros eilės daugianario grafinis atvaizdavimas

```
Daugianario laipsnis: 2
26079.529760779216 * x ^ 0 +
509.6955836700845 * x ^ 1 +
-42.06123138983821 * x ^ 2 +
```

pav. 17 Antros eilės daugianario išraiška

3.3.3 Trečios eilės daugianaris

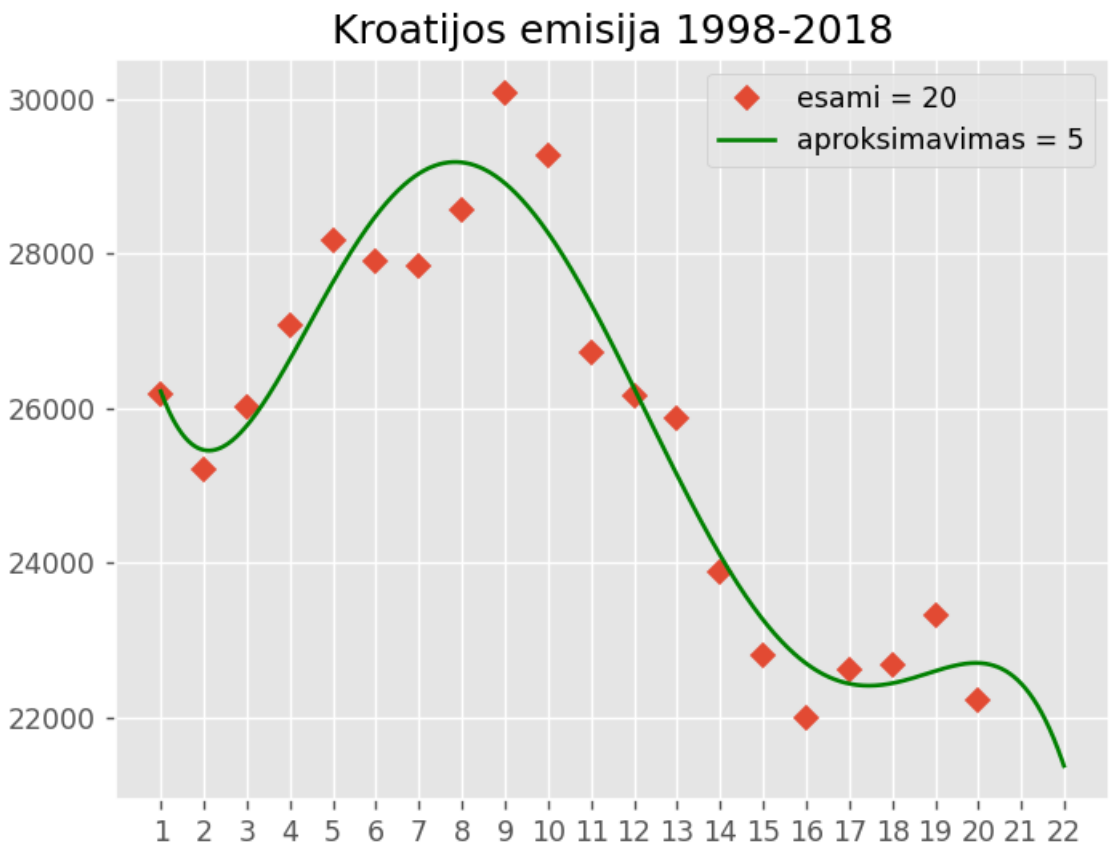


pav. 18 Trečios eilės daugianario grafinis atvaizdavimas

```
Daugianario laipsnis: 3
24385.573951775426 * x ^ 0 +
1739.8093479655627 * x ^ 1 +
-208.1353303117755 * x ^ 2 +
5.827161365681996 * x ^ 3 +
```

pav. 19 Trečios eilės daugianario išraiška

3.3.4 Penktos eilės daugianaris



pav. 20 Penktos eilės daugianario grafinis atvaizdavimas

```
Daugianario laipsnis: 5
26216.965988050237 * x ^ 0 +
-1512.0459343748798 * x ^ 1 +
872.5744550077251 * x ^ 2 +
-127.05175453244914 * x ^ 3 +
6.872278591945076 * x ^ 4 +
-0.12678847175859945 * x ^ 5 +
```

pav. 21 Penktos eilės daugianario išraiška

Išvada: mažiausių kvadratų metodu buvo sudarytos aproksimuojančios kreivės, rezultatai pasitvirtino, esant vis aukštesnei eilei aproksimavimas darosi tikslesnis.

Matas Palujanskas

4. Ketvirta dalis

4.1 Užduoties sąlyga

IV užduotis. Parametrinis aproksimavimas.

Naudodami **parametrinį aproksimavimą Haro bangelėmis** suformuokite ***2 lentelėje*** nurodytos šalies kontūrą. Analizuokite bent 10 detalumo lygių. Pateikite aproksimavimo rezultatus (aproksimuotą kontūro kreivę) ne mažiau kaip 4 skirtinguose lygmenyse. Jei šalis turi keletą atskirų teritorijų (pvz., salų), pakanka analizuoti didžiausią iš jų.

pav. 22 4 dalies sąlyga

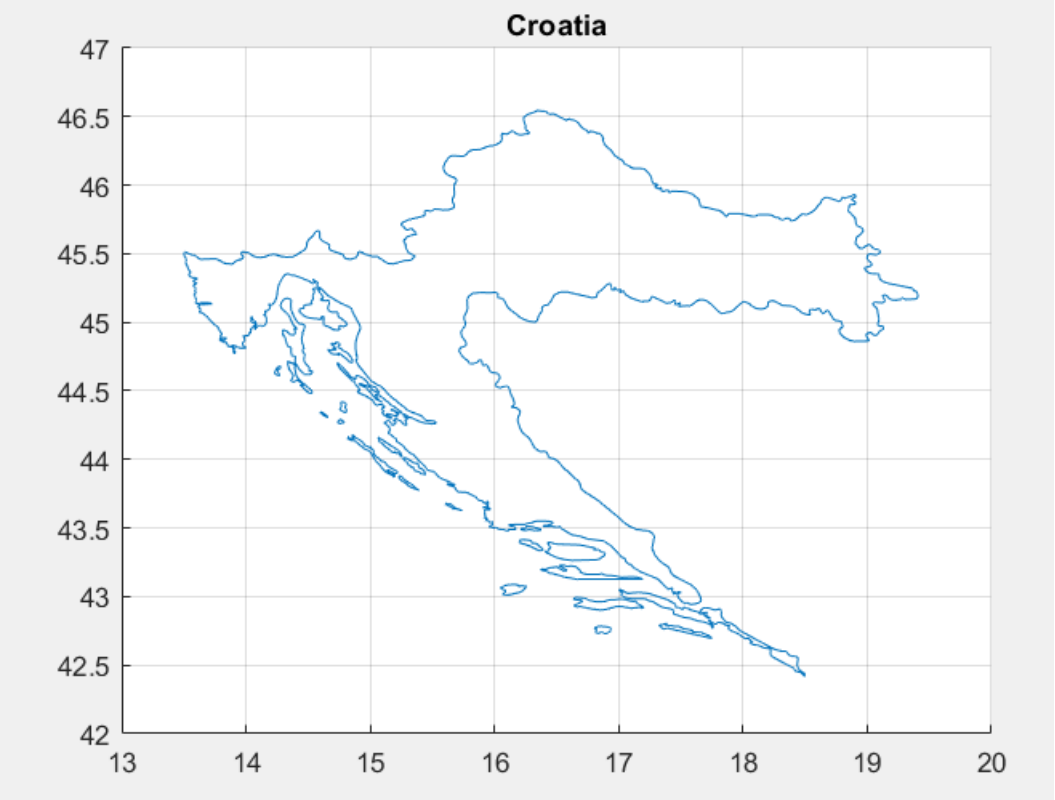
4.2 Sprendimo kodas

Lab3-4.py:

```
import matplotlib.pyplot as plt
import matplotlib
matplotlib.use('TkAgg')
import numpy as np
from interpolation import parametric_interpolation, hermite_interpolation_spline
from lab3_4_data import x_range, y_range

n = 500  # Number of interpolation points
step = 0.1  # Graph's precision

# Reducing interpolation points to selected
t = range(n + 1)
x_range = x_range[::(2359 // n)]
y_range = y_range[::(2359 // n)]

ff = hermite_interpolation_spline(t, x_range)
ff2 = hermite_interpolation_spline(t, y_range)

xx, yy = parametric_interpolation(ff, ff2, np.arange(0, n, step))

# Plot country borders as a line
plt.plot(x_range, y_range, 'r-', label="Šalies ribos")

# Scatter plot for interpolation points
plt.scatter(x_range, y_range, color='blue',  marker='o', label=f"{n} Aproksimavimo
taškai")

plt.title('Kroatija')
plt.legend()
plt.show()
```

4.3 Varianto šalies kontūrai

Gauto 5 varianto šalis yra Kroatija, jos kontūrai iš countries.zip:



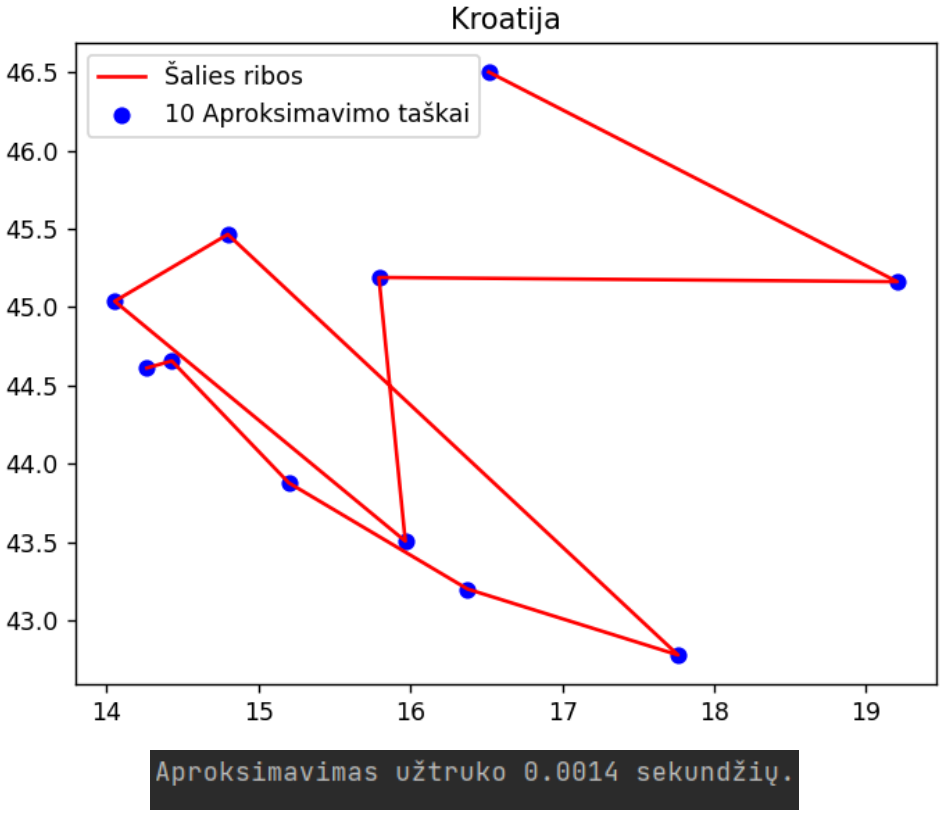
pav. 23 Originalūs Kroatijos kontūrai

4.4 Gauti rezultatai

Tikslas yra gauti kuo tikslesnius šalies kontūrus.

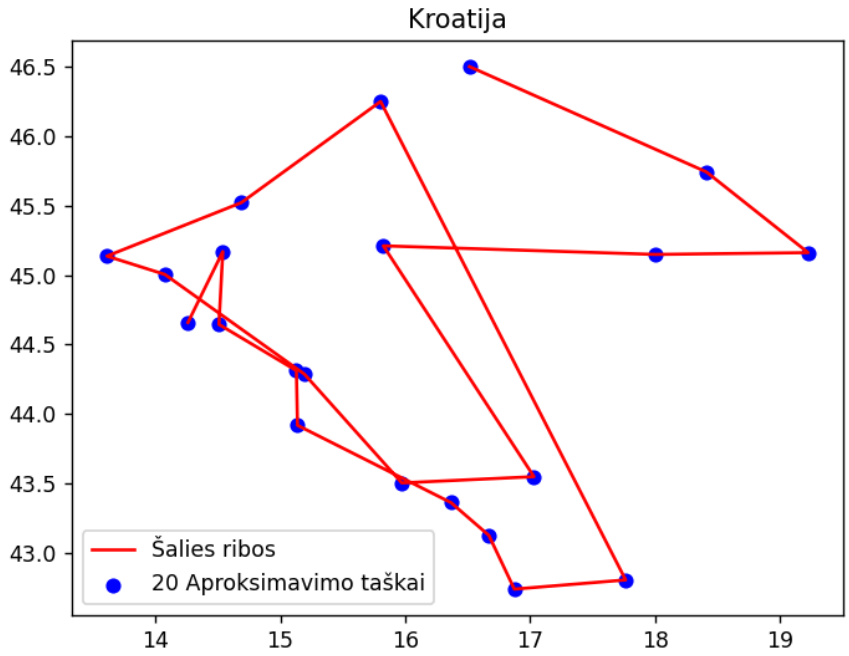
Grafiko tikslumas, pradinis žingsnis yra 0,1.

Pradžioje buvo aproksimuojama su 10 taškų:



pav. 24 Šalies kontūrai naudojant 10 taškų

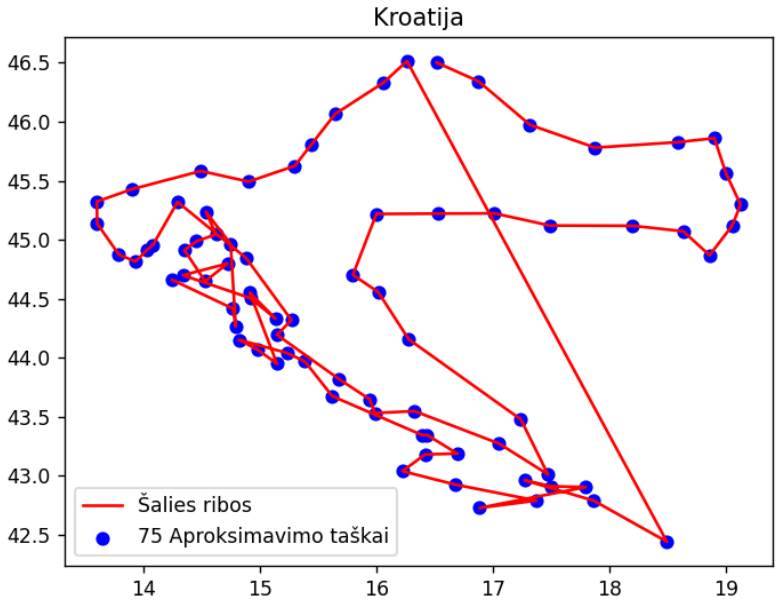
Naudojant 20 taškų su 0.1 žingsniu:




```
Aproksimavimas užtruko 0.0042 sekundžių.
```

pav. 25 Šalies kontūrai naudojant 20 taškų

Naudojant 75 taškus ir žingsnį pakeitus į 0.5:

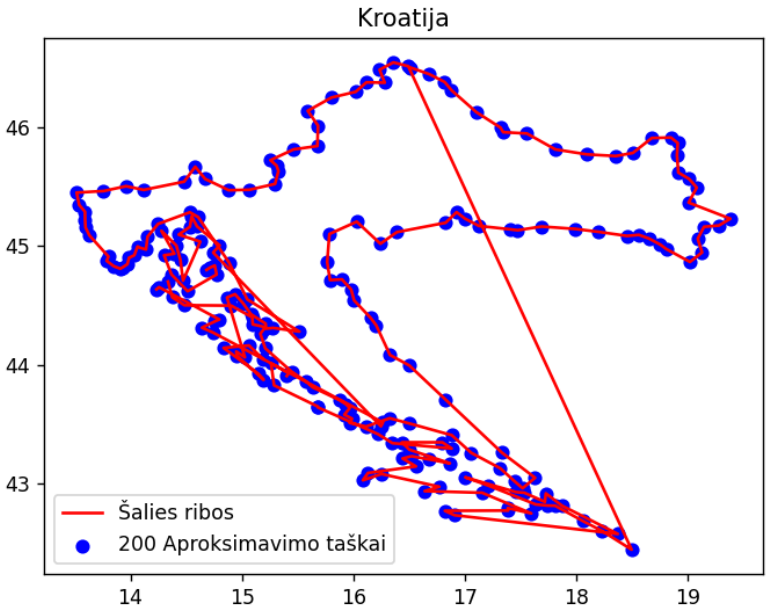



```
Aproksimavimas užtruko 0.0041 sekundžių.
```

pav. 26 Šalies kontūrai naudojant 75 taškus

Ties tokiu taškų skaičiumi galime pastebėti, jog ryškėja šalies kontūrai. Taip pat su didesniu žingsniu programa veikia greičiau.

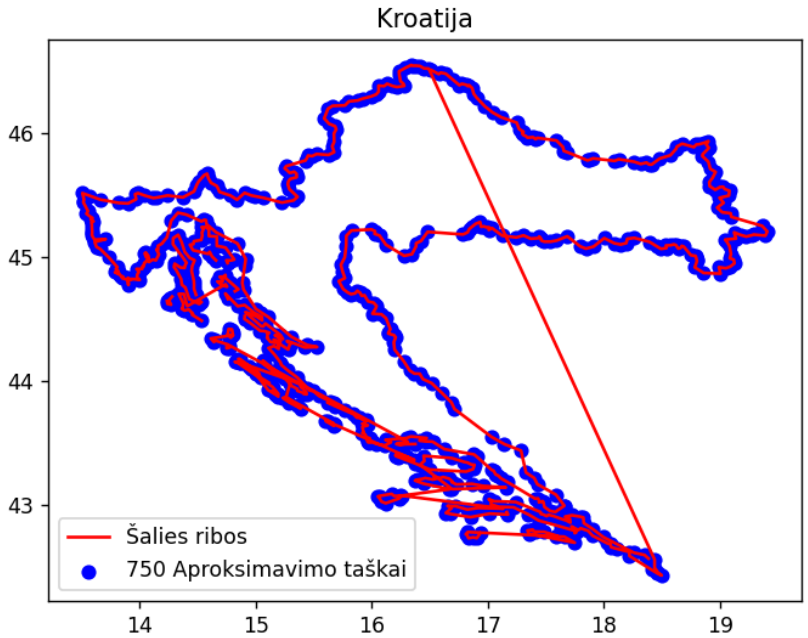
Naudojant 200 taškų ir žingsnį pakeitus į 0.5:




```
Aproksimavimas užtruko 0.0156 sekundžių.
```

pav. 27 Šalies kontūrai naudojant 200 taškų

Naudojant 750 taškų su 0.1 žingsniu:



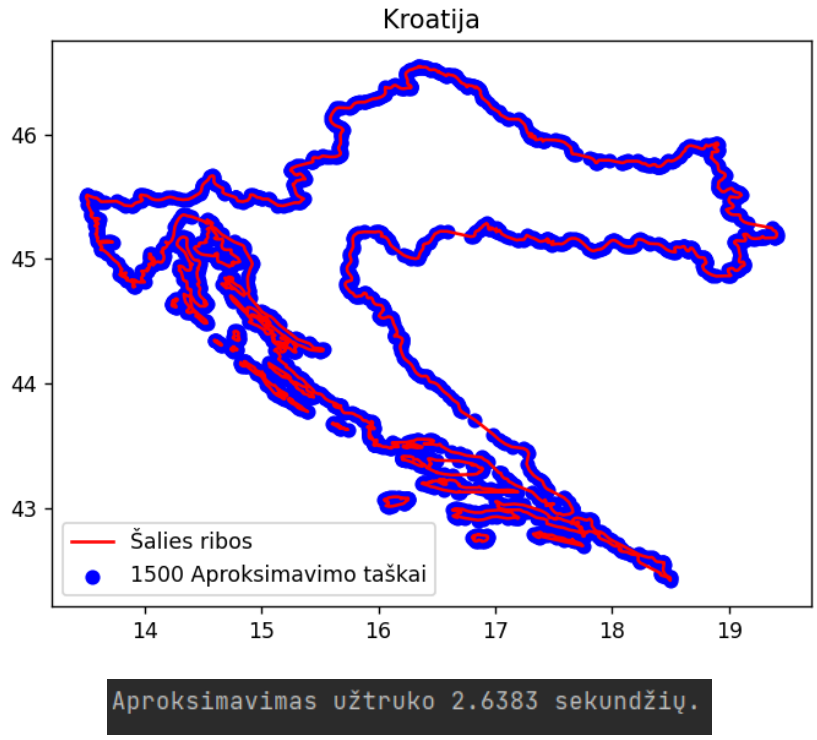

```
Aproksimavimas užtruko 0.7226 sekundžių.
```

pav. 28 Šalies kontūrai naudojant 750 taškų

Su tokiu kiekiu taškų programa jau dirba ilgiau. Galime pastebėti, jog yra jau aiškūs šalies kontūrai, tačiau dar yra netikslumų.

Matas Palujanskas

Naudojant 1500 taškų su 0.1 žingsniu:



pav. 29 Šalies kontūrai naudojant 1500 taškų

Su tokiu taškų kiekiu jau galime aiškiai matyti Kroatijos kontūrus.

Išvada: didėjant taškų kiekiui, buvo tikslinamas ir šalies kontūras, tačiau ilgėjo ir programos vykdomas laikas.

5. Literatūros sąrašas

1. „Skaitiniai metodai ir algoritmai“ „Moodle“ aplinkoje HTTPS://MOODLE.KTU.EDU/COURSE/VIEW.PHP?ID=7639