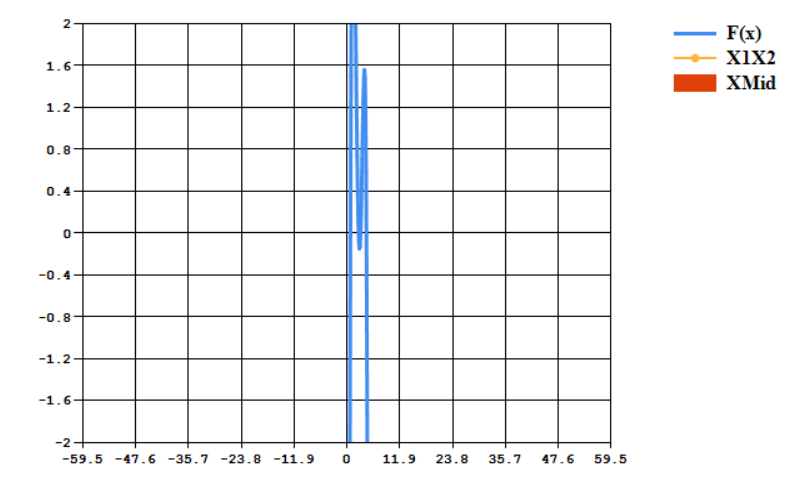
# Netiesinių lygčių sprendimas

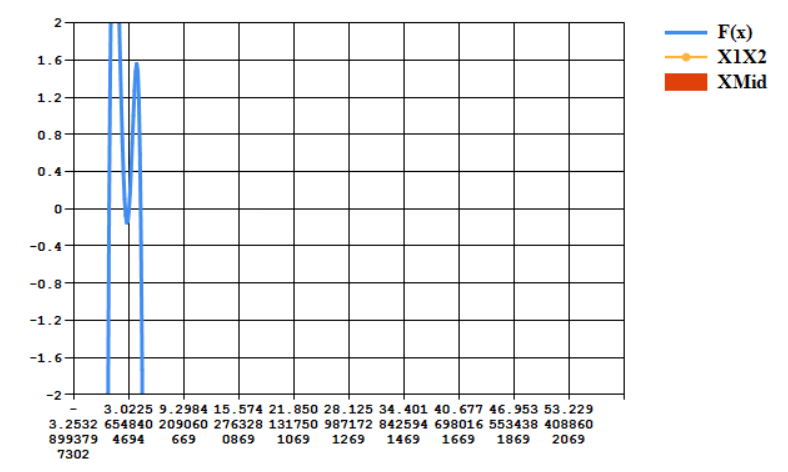
Duotos dvi netiesinės lygtys: daugianaris f(x) = 0 ir transcendentinė funkcija g(x) = 0.

|  |  |
| --- | --- |
| Daugianaris f(x) | Funkcija g(x) |
|  |  |
| Sprendimo metodai: skenavimo, stygų, Niutono(liestinių) | |

## Lygties f(x) = 0 (f(x) – daugianaris) sprendimas

* **Daugianario šaknų intervalo įverčiai**

pav. 1 Daugianario grubaus šaknų intervalo iverčiai. Grafiko dydis yra grubaus šaknų intervalo dydžio



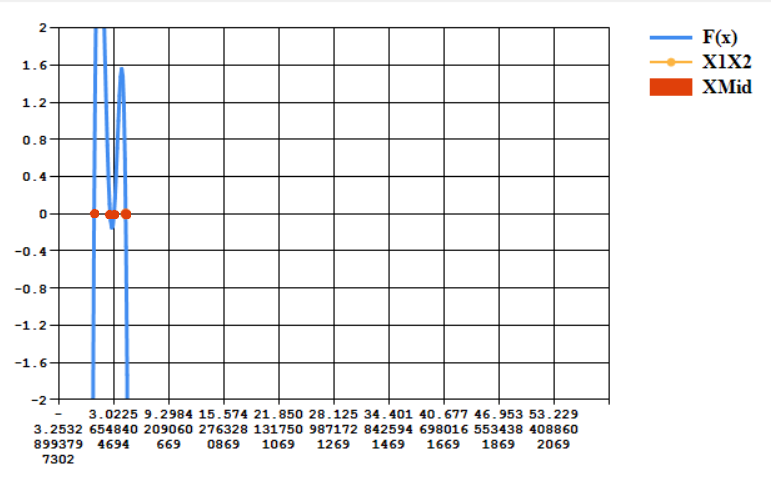
pav. 2 Daugianario tikslesnio šaknų intervalo įverčiai. Grafiko dydis yra tikslesnio šaknų intervalo dydžio

* **1 lentelė.** Šaknų intervalo įverčiai.

|  |  |
| --- | --- |
| Grubus lygties 𝑓(𝑥) = 0 šaknų intervalo įvertis | [-59,5; 59,5 ] |
| Tikslesnis lygties 𝑓(𝑥) = 0 šaknų intervalo įvertis | [-3,25329; 59,505263] |

* **Šaknų atskyrimas skenavimo metodu**

Skenavimas atliekamas intervale [-3,25329; 59,505263], skenavimo žingsnis lygus 1.



*pav.3 Daugianario šaknų atskyrimo intervalai*

**2 lentelė.** Šaknies atskyrimo intervalai.

|  |  |
| --- | --- |
| Intervalo Nr. | Intervalas |
| 1 | [0.71; 0.81] |
| 2 | [2.51; 2.61] |
| 3 | [3.01; 3.11] |
| 4 | [4.31; 4.41] |

* **Šaknų tikslinimas skenavimo, stygų, Niutono (liestinių) metodais**

Tariama, kad 𝑥𝑔 yra šaknis (stabdomi skaičiavimai), jei |𝑓(𝑥𝑔)| < 1𝑒 −6 . Skaičiavimuose naudojamas šaknies tikslumo įvertis |𝑓(𝑥𝑔)|.

**3 lentelė.** Rezultatų lentelė.

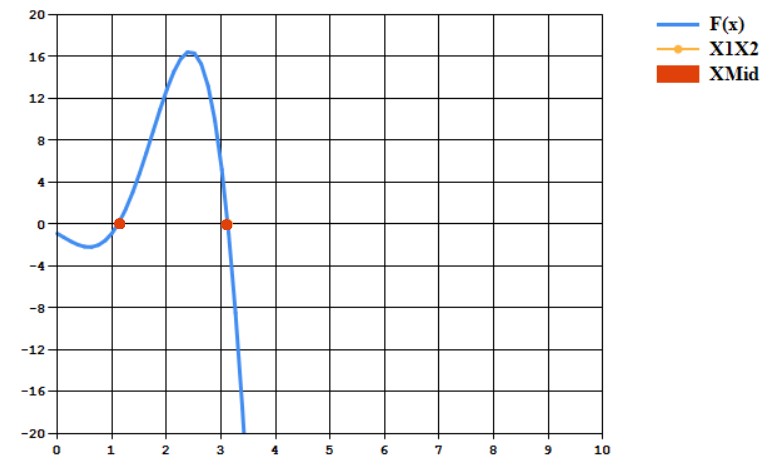
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Skenavimo metodas*** | Pradinis intervalas | Šaknis | Tikslumas | Iteracijų skaičius |
| [0.71; 0.81] | 0.7560098 | -0.0000008 | 32 |
| [2.51; 2.61] | 2.5804280 | 0.0000004 | 25 |
| [3.01; 3.11] | 3.0329570 | -0.0000007 | 29 |
| [4.31; 4.41] | 4.3569200 | 0.0000005 | 24 |
| ***Stygų metodas*** | Pradinis intervalas | Šaknis | Tikslumas | Iteracijų skaičius |
| [0.71; 0.81] | 0.7560099 | 0.0000004 | 4 |
| [2.51; 2.61] | 2.5804285 | -0.0000003 | 5 |
| [3.01; 3.11] | 3.0329569 | -0.0000009 | 4 |
| [4.31; 4.41] | 4.3569200 | 0.0000001 | 5 |
| ***Niutono (liestinių) metodas*** | Pradinis artinys | Šaknis | Tikslumas | Iteracijų skaičius |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Nepavyko iki galo realizuoti niutono metodo ir surasti tinkamų artinių. Pagal turimus rezultatus galima teigti kad f(x) = 0 šaknis randa su mažiausiai iteracijų – stygų metodas. Metodų rezultatai ir tikslumas yra gan panašūs.

## Lygties g(x)=0 (g(x) – transcendentine funkcija) sprendimas

**4 lentelė.** Šaknies atskyrimo intervalai.

|  |  |
| --- | --- |
| Intervalo Nr. | Intervalas |
| 1 | [1.1; 1.2] |
| 2 | [3.1; 3.2] |



pav. 4 Funkcijos šaknų intervalų rėžiai, pavaizduoti grafiškai

**5 lentelė.** Rezultatų lentelė.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Skenavimo metodas*** | Pradinis intervalas | Šaknis | Tikslumas | Iteracijų skaičius |
| [1.1; 1.2] | 1.1151468 | -0.0000007 | 30 |
| [3.1; 3.2] | 3.1248262 | 0.0000009 | 33 |
| ***Stygų metodas*** | Pradinis intervalas | Šaknis | Tikslumas | Iteracijų skaičius |
| [1.1; 1.2] | 1.1151468 | -0.0000005 | 5 |
| [3.1; 3.2] | 3.1248262 | 0.0000001 | 5 |
| ***Niutono (liestinių) metodas*** | Pradinis artinys | Šaknis | Tikslumas | Iteracijų skaičius |
| 1 | 1.1151469 | 0.0000000 | 4 |
|  |  |  |  |

Dėl nepilnai implementuoto Niutono algoritmo surasta tik viena šaknis. Iš gautų rezultų galima teigti, kad g(x) = 0 šaknų radimui mažiausiai iteracijų reikia naudojant Niutono metodą. Taip pat Niutono metodas yra tiksliausias iš duotų.

## Sąlyginio uždavinio sprendimas

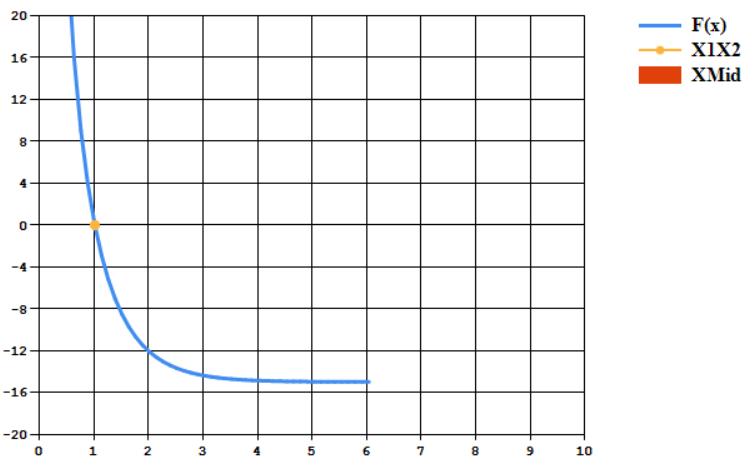
**Sąlyga.**

Vertikaliai į viršų iššauto objekto greitis užrašomas dėsniu , čia g = 9,8 m/s2, pradinis greitis v0, objekto masė m. Koks pasipriešinimo koeficientas c veikia objektą, jei žinoma, kad po t1 laiko nuo iššovimo jo greitis lygus v1?

V0 = 50, m = 2, t1 = 3, v1 = 14

Sudaroma funkcija , ir ieškoma kur funkcija kerta X ašį.

Gauta šaknis: 1.0195358

pav. 5 Sąlyginio uždavinio grafikas ir jo šaknys

# Išvados

Laboratorinio darbo metu buvo analizuojama algebrinės lygties su vienu nežinomuoju sprendimo etapai – šaknų atskyrimo ir jų tikslinimo uždaviniai. Buvo įgyvendinti skenavimo, stygų, Niutono (nepilnas) metodai. Niutono metodo įgyvendinimas buvo sunkiausiais, tačiau šaknis suranda su dideliu tikslumu.

# Programos tekstai

**Skenavimo metodas:**

private void button7\_Click(object sender, EventArgs e)

{

ClearForm();

double x = 0;

int i = 0;

switch (comboBox1.SelectedIndex)

{

case 0:

PreparareForm(0, 10, -5, 5);

break;

case 1:

PreparareForm(0, 10, -20, 20);

break;

case 2:

PreparareForm(0, 10, -20, 20);

x = 0.01;

i = 1;

break;

}

richTextBox1.AppendText("Iteracija x F(x) x1 x2 F(x1) F(x2) \n");

Fx = chart1.Series.Add("F(x)");

Fx.ChartType = SeriesChartType.Line;

for (; i < 50; i++)

{

Fx.Points.AddXY(x, F(x)); x = x + (2 \* Math.PI) / 50;

}

Fx.BorderWidth = 3;

X1X2 = chart1.Series.Add("X1X2");

X1X2.MarkerStyle = MarkerStyle.Circle;

X1X2.MarkerSize = 8;

X1X2.ChartType = SeriesChartType.Point;

X1X2.ChartType = SeriesChartType.Line;

XMid = chart1.Series.Add("XMid");

XMid.MarkerStyle = MarkerStyle.Circle;

X1X2.ChartType = SeriesChartType.Point;

X1X2.ChartType = SeriesChartType.Line;

XMid.MarkerSize = 8;

var thing = intervals.Dequeue();

x1 = thing.Item1;

x2 = thing.Item2;

timer5.Enabled = true;

timer5.Interval = 50;

timer5.Start();

}

private void timer5\_Tick(object sender, EventArgs e)

{

xtemp = x1 + stepSize;

if (Math.Abs(F(xtemp)) > 1e-6 & iii <= N)

{

X1X2.Points.Clear();

X1X2.Points.AddXY(x1, 0);

X1X2.Points.AddXY(xtemp, 0);

richTextBox1.AppendText($" {iii,6:d} {xtemp,12:f7} {F(xtemp),12:f7} {x1,12:f7} {x2,12:f7} {F(x1),12:f7} {F(x2),12:f7}\n");

if (Math.Sign(F(x1)) != Math.Sign(F(xtemp)))

{

stepSize /= 10;

}

else

{

x1 += stepSize;

}

iii = iii + 1;

}

else

{

richTextBox1.AppendText($" {iii,6:d} {xtemp,12:f7} {F(xtemp),12:f7} {x1,12:f7} {x2,12:f7} {F(x1),12:f7} {F(x2),12:f7}\n");

richTextBox1.AppendText("Skaičiavimai baigti");

if (intervals.Any())

{

var thing = intervals.Dequeue();

x1 = thing.Item1;

x2 = thing.Item2;

stepSize = 0.1;

}

else

{

timer5.Stop();

}

}

}

**Stygų metodas:**

private void button6\_Click(object sender, EventArgs e)

{

ClearForm();

double x = 0;

int i = 0;

switch (comboBox1.SelectedIndex)

{

case 0:

PreparareForm(0, 10, -5, 5);

break;

case 1:

PreparareForm(0, 10, -20, 20);

break;

case 2:

PreparareForm(0, 10, -20, 20);

x = 0.01;

i = 1;

break;

}

richTextBox1.AppendText("Iteracija x F(x) x1 x2 F(x1) F(x2) \n");

Fx = chart1.Series.Add("F(x)");

Fx.ChartType = SeriesChartType.Line;

for (; i < 50; i++)

{

Fx.Points.AddXY(x, F(x)); x = x + (2 \* Math.PI) / 50;

}

Fx.BorderWidth = 3;

X1X2 = chart1.Series.Add("X1X2");

X1X2.MarkerStyle = MarkerStyle.Circle;

X1X2.MarkerSize = 8;

X1X2.ChartType = SeriesChartType.Point;

X1X2.ChartType = SeriesChartType.Line;

XMid = chart1.Series.Add("XMid");

XMid.MarkerStyle = MarkerStyle.Circle;

X1X2.ChartType = SeriesChartType.Point;

X1X2.ChartType = SeriesChartType.Line;

XMid.MarkerSize = 8;

var thing = intervals.Dequeue();

x1 = thing.Item1;

x2 = thing.Item2;

timer4.Enabled = true;

timer4.Interval = 500;

timer4.Start();

}

private void timer4\_Tick(object sender, EventArgs e)

{

var cof = Math.Abs(F(x1) / F(x2));

xtemp = (x1 + cof \* x2) / (1 + cof);

if (Math.Abs(F(xtemp)) > 1e-6 & iii <= N)

{

X1X2.Points.Clear();

XMid.Points.Clear();

X1X2.Points.AddXY(x1, F(x1));

X1X2.Points.AddXY(x2, F(x2));

XMid.Points.AddXY(xtemp, 0);

richTextBox1.AppendText($" {iii,6:d} {xtemp,12:f7} {F(xtemp),12:f7} {x1,12:f7} {x2,12:f7} {F(x1),12:f7} {F(x2),12:f7}\n");

if (Math.Sign(F(x1)) != Math.Sign(F(xtemp)))

{

x2 = xtemp;

}

else

{

x1 = xtemp;

}

iii = iii + 1;

}

else

{

richTextBox1.AppendText($" {iii,6:d} {xtemp,12:f7} {F(xtemp),12:f7} {x1,12:f7} {x2,12:f7} {F(x1),12:f7} {F(x2),12:f7}\n");

richTextBox1.AppendText("Skaičiavimai baigti");

if (intervals.Any())

{

var thing = intervals.Dequeue();

x1 = thing.Item1;

x2 = thing.Item2;

}

else

{

timer4.Stop();

}

}

}

**Niutono (liestinių metodas):**

private void button8\_Click(object sender, EventArgs e)

{

ClearForm();

double x = 0;

int i = 0;

switch (comboBox1.SelectedIndex)

{

case 0:

PreparareForm(0, 10, -5, 5);

x1 = 1;

break;

case 1:

PreparareForm(0, 10, -20, 20);

x1 = 1;

break;

case 2:

PreparareForm(0, 10, -20, 20);

x1 = 1.5;

x = 0.01;

i = 1;

break;

}

richTextBox1.AppendText("Iteracija x F(x) x1 x2 F(x1) F(x2) \n");

Fx = chart1.Series.Add("F(x)");

Fx.ChartType = SeriesChartType.Line;

for (; i < 50; i++)

{

Fx.Points.AddXY(x, F(x)); x = x + (2 \* Math.PI) / 50;

}

Fx.BorderWidth = 3;

X1X2 = chart1.Series.Add("X1X2");

X1X2.MarkerStyle = MarkerStyle.Circle;

X1X2.MarkerSize = 8;

X1X2.ChartType = SeriesChartType.Point;

X1X2.ChartType = SeriesChartType.Line;

XMid = chart1.Series.Add("XMid");

XMid.MarkerStyle = MarkerStyle.Circle;

X1X2.ChartType = SeriesChartType.Point;

X1X2.ChartType = SeriesChartType.Line;

XMid.MarkerSize = 8;

timer6.Enabled = true;

timer6.Interval = 500;

timer6.Start();

}

private void timer6\_Tick(object sender, EventArgs e)

{

xtemp = -F(x1) / dF(x1);

x1 += xtemp;

xtemp = x1;

if (Math.Abs(F(x1)) > 1e-6 & iii <= N)

{

X1X2.Points.Clear();

XMid.Points.Clear();

XMid.Points.AddXY(x1, 0);

richTextBox1.AppendText($" {iii,6:d} {xtemp,12:f7} {F(xtemp),12:f7} {x1,12:f7} {x2,12:f7} {F(x1),12:f7} {F(x2),12:f7}\n");

iii = iii + 1;

}

else

{

richTextBox1.AppendText($" {iii,6:d} {xtemp,12:f7} {F(xtemp),12:f7} {x1,12:f7} {x2,12:f7} {F(x1),12:f7} {F(x2),12:f7}\n");

richTextBox1.AppendText("Skaičiavimai baigti");

timer6.Stop();

}

}

**Papildomos funkcijos:**

private double F(double x)

{

switch (comboBox1.SelectedIndex)

{

case 0: return - 0.95 \* Math.Pow(x, 4)

+ 10.19 \* Math.Pow(x, 3)

- 37.83 \* Math.Pow(x, 2)

+ 55.58 \* x

- 24.49;

case 1: return Math.Sin(x) \* (x \* x - 1) \* (x + 3) - 0.9;

case 2: return 50 \* Math.Pow(Math.E, -x \* 3 / 2)

+ (19.6 / x \* Math.Pow(Math.E, -x \* 3 / 2) - 1)

- 14;

}

return 0;

}

private double dF(double x)

{

switch (comboBox1.SelectedIndex)

{

case 0:

return 3.8 \* Math.Pow(x, 3)

- 30.57 \* Math.Pow(x, 2)

+ 75.66 \* x

- 55.58;

case 1:

return Math.Sin(x) \* (x \* x - 1)

+ Math.Cos(x) \* (x \* x - 1) \* (x + 3)

+ 2 \* x \* Math.Sin(x) \* (x + 3);

case 2:

return (Math.Pow(Math.E, -x \* 3 / 2) \* (98 \* Math.Pow(Math.E, x \* 3 / 2)) - 375 \* x \* x - 147 \* x - 98) / 5 \* x \* x;

}

return 0;

}

private Queue<Tuple<double, double>> int1 = new Queue<Tuple<double, double>>();

private Queue<Tuple<double, double>> int2 = new Queue<Tuple<double, double>>();

private Queue<Tuple<double, double>> int3 = new Queue<Tuple<double, double>>();

private Queue<Tuple<double, double>> intervals

{

get

{

switch (comboBox1.SelectedIndex)

{

case 0: return int1;

case 1: return int2;

case 2: return int3;

default: return null;

}

}

}

private void FillIntervals()

{

int1.Clear();

int2.Clear();

int3.Clear();

comboBox1.SelectedIndex = 0;

x1 = 0.01;

x2 = 10;

var xTemp = x1;

for (x1 += stepSize; x1 < x2; x1 += stepSize)

{

if (Math.Sign(F(x1)) != Math.Sign(F(xTemp)))

{

int1.Enqueue(new Tuple<double, double>(xTemp, x1));

}

xTemp = x1;

}

comboBox1.SelectedIndex = 1;

x1 = 0;

x2 = 5;

xTemp = x1;

for (x1 += stepSize; x1 < x2; x1 += stepSize)

{

if (Math.Sign(F(x1)) != Math.Sign(F(xTemp)))

{

int2.Enqueue(new Tuple<double, double>(xTemp, x1));

}

xTemp = x1;

}

comboBox1.SelectedIndex = 2;

x1 = 0.5;

x2 = 10;

xTemp = x1;

for (x1 += stepSize; x1 < x2; x1 += stepSize)

{

if (Math.Sign(F(x1)) != Math.Sign(F(xTemp)))

{

int3.Enqueue(new Tuple<double, double>(xTemp, x1));

}

xTemp = x1;

}

}